AD-A030 956

A Simulator Program for TSS

Michigan Univ Ann Arbor

Dec 75

BEST SELLERS

FROM NATIONAL TECHNICAL INFORMATION SERVICE

NTIS

Product Liability Insurance: Assessment of Related Problems and Issues. Staff Study
PB-252 204/PAT 181 p PC$9 50/MF$3 00

Evaluation of Home Solar Heating System
UCRL-51 711/PAT 154 p PC$6 75/MF$3 00

Developing Noise Exposure Contours for General Aviation Airports
ADA-023 429/PAT 205 p PC$7 75/MF$3.00

Cooling Tower Environment, 1974. Proceedings of a Symposium Held at the University of Maryland Adult Education Center on Mar. 4-6, 1974
CONF-74 0302/PAT 648 p PC$13 60/MF$3.00

Biological Services Program. Fiscal Year 1975
PB-251 738/PAT 52 p PC$4.50/MF$3 00

An Atlas of Radiation Histopathology
TID-26-676/PAT 234 p PC$7 60/MF$3.00

Federal Funding of Civilian Research and Development. Vol. 1. Summary
PB-251 266/PAT 61 p PC$4 50/MF$3.00

Federal Funding of Civilian Research and Development. Vol. 2. Case Studies
PB-251 683/PAT 336 p PC$10 00/MF$3 00

Handbook on Aerosols
TID-26-608/PAT 141 p PC$6 00/MF$3 00

for the Assessment of Ocean Outfalls
ADA-023 514/PAT 34 p PC$4.00/MF$3 00

Guidelines for Documentation of Computer Programs and Automated Data Systems
PB-250 867/PAT 54 p PC$4 50/MF$3 00

NOx Abatement for Stationary Sources in Japan
PB-250 586/PAT 116 p PC$5 50/MF$3 00

U.S. Coal Resources and Reserves
PB-252 752/PAT 16 p PC$3 50/MF$3 00

Structured Programming Series. Vol. XI. Estimating Software Project Resource Requirements
ADA-016 416/PAT 70 p PC$4.50/MF$3 00

Assessment of a Single Family Residence Solar Heating System in a Suburban Development Setting
PB-246 141/PAT 244 p PC$8 00/MF$3.00

Technical and Economic Study of an Underground Mining, Rubblization, and in Situ Retorting System for Deep Oil Shale Deposits. Phase I Report
PB-249 344/PAT 223 p PC$7 75/MF$3.00

A Preliminary Forecast of Energy Consumption Through 1985
PB-251 445/PAT 69 p PC$4 50/MF$3 00

HOW TO ORDER

When you indicate the method of payment, please note if a purchase order is not accompanied by payment, you will be billed an addition $5 00 *ship and bill* charge And please include the card expiration date when using American Express

Normal delivery time takes three to five weeks It is vital that you order by number or your order will be manually filled, insuring a delay You can opt for *airmail delivery* for a $2.00 charge per item Just check the *Airmail Service* box If you're really pressed for time, call the NTIS Rush Order Service (703) 557-4700 For a $10 00 charge per item, your order will be airmailed within 48 hours. Or, you can pick up your order in the Washington Information Center & Bookstore or at our Springfield Operations Center within 24 hours for a $6 00 per item charge

You may also place your order by telephone or TELEX The order desk number is (703) 557-4650 and the TELEX number is 89 9405

Whenever a foreign sales price is NOT specified in the listings, all foreign buyers must add the following charges to each order: $2 50 for each paper copy, $1.50 for each microfiche; and $10 00 for each Published Search.

Thank you for your interest in NTIS. We appreciate your order

METHOD OF PAYMENT

☐ Charge my NTIS deposit account no _____

☐ Purchase order no _____

☐ Check enclosed for $_____

☐ Charge to my American Express Card account number

Card expiration date _____

Signature _____

☐ Airmail Services requested

NAME _____

ADDRESS _____

CITY STATE ZIP _____

Item Number	Quantity		Unit Price*	Total Price
	Paper Copy (PC)	Microfiche (MF)		
			Sub Total	
			Additional Charge	
			Enter Grand Total	

All Prices Subject to Change

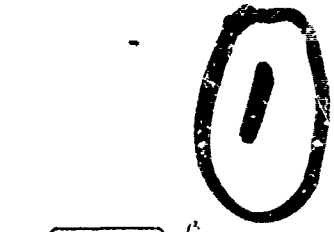
11/76

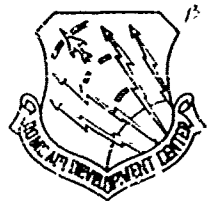
Clip and mail to

NTIS

National Technical Information Service
U.S. DEPARTMENT OF COMMERCE
Springfield, Va. 22161
(703) 557-4650 TELEX 89-9405

296068





RADC-TR-75-313
Interim Report
December 1975

A SIMULATOR PROGRAM FOR TSS

The University of Michigan

Approved for public release;
distribution unlimited.

SUBJECT TO CHANGE

This report has been reviewed by the RADC Information Office (OI) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

This report has been reviewed and approved for publication.

APPROVED: Rocco F Iuorno

ROCCO F. IUORNO
Project Engineer

APPROVED: Robt D Krutz

ROBERT D. KRUTZ, Col., USAF
Chief, Information Sciences Division

FOR THE COMMANDER: John P. Huss

JOHN P. HUSS
Acting Chief, Plans Office

Do not return this copy. Retain or destroy.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1 REPORT NUMBER: RADC-TR-75-313	2. GOVT ACCESSION NO.
3 RECIPIENT'S CATALOG NUMBER	
4 TITLE (and Subtitle): A SIMULATOR PROGRAM FOR TSS	5 TYPE OF REPORT & PERIOD COVERED: Interim Report, 1 Jul 72 - 30 Sep 75
6 PERFORMING ORG. REPORT NUMBER: N/A	
7. AUTHOR(s): Jamshed D. Mulla under the direction of Professor Keki B. Irani	8 CONTRACT OR GRANT NUMBER(s): F30602-73-C-0001
9. PERFORMING ORGANIZATION NAME AND ADDRESS: The University of Michigan/Department of Electrical & Computer Engineering, Ann Arbor MI 48104	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: 62702F, 55810211
11. CONTROLLING OFFICE NAME AND ADDRESS: Rome Air Development Center (ISIS), Griffiss AFB NY 13441	12. REPORT DATE: December 1975
	13 NUMBER OF PAGES: 178
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office): Same	15. SECURITY CLASS. (of this report): UNCLASSIFIED
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE: N/A

16 DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17 DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

Same

18 SUPPLEMENTARY NOTES

RADC Project Engineer:
Rocco F. Iuorno (ISIS)

19 KEY WORDS (Continue on reverse side if necessary and identify by block number)

Simulation	Time-Sharing System
Modeling	Event Oriented
Simscript II	Implementation of Model

20 ABSTRACT (Continue on reverse side if necessary and identify by block number)

This report describes the computer program and its implementation which simulates the Honeywell Time Sharing System (Version 8) operating on the Honeywell 635. The program (model) simulates the behavior of Subsystem programs and the efforts of the scheduling policies of the Time Sharing System. The simulation program is written in Simscript II programming language.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

Table of Contents

Figures

Appendices

1. Introduction

The purpose of this report is to describe a program written to simulate the Honeywell Time Sharing System (Version 8) operating on the Honeywell 635 computer.

The model includes, in great detail, sections of the TSS Allocator as well as the TSS Executive, the Derail Processor and Line Service. The program simulates the behavior of Subsystem programs and the effects of the scheduling policies of TSS.

The simulation program being described here is written in the Simscript II programming language. Simscript is a high level, event oriented simulation language with several features that are very helpful for this particular use.

The program was written and developed at the Systems Engineering Laboratory at the University of Michigan and was debugged and validated on a Honeywell 3000 system at the Rome Air Development Center, Rome, New York.

2. Event and Routine Summary

As mentioned before, Simscript is an event oriented language and hence the program is made up of several small events and routines, each corresponding to a subroutine or block of code of TSS. To ease understanding and debugging the names of all routines and events have been preserved from the actual system program listings as far as possible.[1]

The events and routines in the simulation program can be classified into six categories. The following is a list of the categories and the routines comprising each one with a brief description of their functions.

2.1 Initialization

INIT - Reads input parameters and values. Initializes data structures. Schedules events for termination and diagnostic printing.

MAIN - Starts simulation.

2.2 Diagnostic and Output

PDIAG - Controls printing of diagnostic information as per input parameters.

REINIT - Reinitializes statistic counters after given simulation period.

CORSAMP - Samples values of program size, hole size and used core for collecting statistics.

MQPRINT - Prints memory queue.

CQPRINT - Prints processor queue.

STERM - Terminates simulation.

---

[1] In certain cases where two names have the same first four characters, the names have been transformed due to a Simscript requirement that the first four characters of all names be unique. e.g. ALLCC1 and ALLCC2 are renamed as 1ALLCC and 2ALLCC.

OUTPUT - Prints simulation results and statistics.

SNAP.R - Prints debugging and diagnostic information in case the simulation terminates abnormally.

2.3 Allocator Routines

ALLOCI - Main entry point to the TSS allocator and the process or allocator process (PAP).

MAP - Memory allocator process. Selects programs for memory allocation and swap.

SDP, SDP3, SDP4, SDP5, SDP6, SDP7 - Swap decision processors. Find programs for regular and force swaps, handle urgent user logic, core fence for urgent user and TSS swap area size control.

SPMACT - Special memory action processor for changing TSS swap area size.

MBA, MBA3 - Memory buffer allocator process. Attempts core allocation for programs and manages memory map.

MBD - Memory buffer deallocator. Returns blocks of memory to available storage pool in memory map.

MMV - Memory map verification process. Also accumulates some core statistics.

SWOUT - Swap out routine.

SWIN - Swap in routine.

SWPLD - Schedules courtesy call for swap.

SSFINI - Terminates subsystem execution.

BUFDMP - Adjusts subsystem status for keyboard input and output operations.

KIOSRT - Schedules keyboard I/O courtesy calls.

START - Introduces new subsystems into the system and initializes

subsystem attributes.

SACT - Takes subsystem accounting for statistics.

ATCHG - Changes subsystem state times.

EXENTR - Entry to allocator via an interrupt.

EXEACT - Accounting routine after return from subsystem dispatch.

2.4 Derail Routines

KONDRL - Keyboard input derail.

KOTDRL - Keyboard output derail.

DRLDIO - Disk I/O derail.

DRLRET - Subsystem return (termination) derail.

2.5 Courtesy Calls

KIOCC - Keyboard I/O courtesy call.

DIOCC - Disk I/O courtesy call.

1ALLCC - Swap out courtesy call.

2ALLCC - Swap in courtesy call.

2.6 Line Service

LINSV - Controls TSS action during relinquish to GCOS and TSS idle. Also schedules new program arrivals into TSS.

3. Data Structures

Three major TSS data structures are modelled in the program exactly as they appear in the real system. These are the processor and memory queues and the memory map. In addition, the model also has a disk I/O queue and a queue of scheduled courtesy calls. The purpose of these is discussed below. Finally, a program in TSS is modelled by means of an entity called a UST. (The name UST was chosen because the entity closely resembles an entry in the User Status Table.)

3.1 The UST Entity

Each active program in the system is represented by a temporary entity called a UST. An entity is created when a new job arrives and is destroyed when it terminates.

The UST entity has a number of attributes some of which are similar to those found in the TSS user status table and others which control the activity of the program.

The following is a list of the attributes of the UST entity. The attributes on the left are those that also appear in the TSS user status table and have the same names as the original variables. (Note: The digits 1 and 2 appended to a variable name signify the upper and lower halves of that word respectively.) The attributes on the right are used to control the simulation of the program that the UST represents. A description of each attribute is given in Appendix 1.

UST Attributes

FK19	CHKCPU
FL18	JOBNO
FL19	KILL
FL21	NXTDIO
FL22	NXTKIN
FL23	NXTKOUT
FL24	OUTCC
FL34	DIOIAT
LSIZE	
LSPTS	KIIAT

LSTIO KOIAT

LTC21, LTC22

LTC31, LTC32

LTCW

LTIN

LTM0

LTM1

LTM2

LTM3

LTM4

LTM5

LTMRS

LTMWT

The UST entity contains the entire description of a program in the system. It is filed in the different system queues in exactly the same manner as the entries in the TSS user status table are linked into queues.

Usually, attributes of entities are referred to in Simscript by the expression:

attribute (entity pointer)

to denote the particular entity whose attribute is being examined. However, the entity pointer can be omitted and the default global entity name 'UST' is assumed. Throught most of the program such a scheme is used to refer to attributes since the global variable 'UST' always contains the pointer to the UST on which the allocator is presently working.

3.2 The Processor Queue - CPUQ

The TSS processor queue consists of a linked list of user status table entries. In the program, this queue is modelled as a simple ordered Simscript set called CPUQ. UST entities are filed last in the set and the first entity is always picked for allocation.

3.3 The Memory Queue - MEMQ

The TSS memory queue is a linked list of UST entries ordered by increasing program sizes. This is modelled in the program as a set MEMQ whose members (UST entities) are ranked by low LSIZE attributes. When new UST's are filed in MEMQ, Simscript automatically links them so that the increasing order of the LSIZE attribute is preserved.

3.4 The Memory Map

The memory map data structure is almost identical to its TSS counterpart. The map consists of a doubly linked list of blocks each of which represents one program present in core.

The memory map consists of five vectors: SJOB, SHOLE, SUC, PRED and IDPTR. Each entry in the map is represented by a corresponding element from each of the above vectors. The vectors have the following functions:

SJOB(I) - contains the program size of the program (in units of 1024 words).

SHOLE(I) - contains the size of the "hole" (unused portion of core) following the program in core (in units of 1024 words).

SUC(I) - poincer to the entry in the map succeeding this entry.

PRED(I) - pointer to the entry in the map preceeding this entry.

IDPTR(I) - contains the identification number attribute (JOBNO) of

the UST entity that corresponds to this entry in the memory map.

The first entry in the memory map is always a dummy entry which belongs to no program but is kept merely to hold the hole size at the top of TSS core. The variables HEAD and TAIL point to the first and last entries in the map respectively. A possible memory map configuration is shown in Figure 1 below.

I	SJOB	SHOLE	SUC	PRED	IDPTR
1	0	10	3	0	0
2					
3	10	2	7	1	56
4					
5					
6	11	15	0	7	67
7	22	0	6	3	24

Figure 1. Memory Map Configuration

In the example of Figure 1 the values of HEAD and TAIL would be 1 and 6 respectively.

3.5 The Disk I/O Queue - DIOQ

Since no data is obtained on the service time of the disk units the program uses an artificial disk I/O queue to simulate the handling of disk I/O operations.

The disk I/O queue is a fifo queue and is modelled with a single exponential server. The formula for obtaining the exponential server rate from the measured values of the disk I/O queue and disk I/O operation arrival rate is described in Reference 1.

3.6 The Courtesy Call Queue - CCQ

The courtesy call queue is strictly for the internal use of the model and does not exist in the real system. The purpose of this queue is to accurately simulate the arrival of scheduled courtesy calls when they interrupt the execution of a subsystem program.

In the actual system a courtesy call would generate an interrupt to TSS. In the model, this interrupt is generated with the help of the CCQ.

The CCQ consists merely of a list of all future courtesy calls ordered by increasing arrival times. When a new courtesy call is scheduled it is entered in the queue at the appropriate position depending on its arrival time. When a courtesy call arrives, it is removed from the CCQ. Note that at this time it should be at the head of CCQ and the simulation time will be equal to its arrival time.

When the TSS allocator dispatches the CPU to a subsystem, it checks to see if a courtesy call would be the next event to interrupt the subsystem. This is done by checking the arrival time of the first entry in the CCQ. If that is the case, an interrupt and consequent

entry into the allocator is scheduled immediately following the courtesy call.

4. Input Parameters

The input to the simulation program is divided into three categories as follows. For each category the input parameters are given in the order in which they must appear in the input stream. The designations (R) and (I) specify the mode of the variable as real or integer respectively. All time values are in milliseconds and all memory values are in units of 1024 words unless otherwise stated.

4.1 Diagnostic and Control

These input values control the execution of the simulation program and the printing of diagnostic comments. The form of messages printed is described fully in Section 5.6.

STOPTIME - (R) Value of simulated time at which the simulation is to be terminated.

DEVS - (I) Event diagnostics switch. If this switch is set to 1, a message is printed. If no diagnostics are required this switch must be 0.

DRS - (I) This switch is similar to DEVS but is for allocator routines.

DQS - (I) If this switch is non-zero, the MEMQ or CPUQ are displayed whenever that queue is changed.

DMS - (I) If this switch is non-zero, the memory map is displayed every time it is changed.

DSCS - (I) If this switch is non-zero, a message is printed every

time the TSS swap area size is changed, and a sample of core utilization is displayed at every call of routine MMV.

DUS - (I) If this switch is non-zero a message is printed for every urgent user detected and every force swap performed.

DKS - (I) If this switch is non-zero a message is printed for each key I/O derail and courtesy call.

DSS - (I) If this switch is non-zero, a message is printed at the start and termination of every subsystem.

DBEGIN - (R) Simulation time at which diagnostic printing is to begin.

DDUR - (R) Duration of diagnostic printing.

STREAM(1) . . . STREAM(10) - (I) These ten integer numbers are used as seeds for the ten Simscript pseudo-random number generators.

4.2 TSS Parameters

These input values consist of various TSS parameters that are used by the allocator for decision making. These parameters are described more fully in Reference 2, TSS Executive SMD. The values of each parameter in the system can be obtained from the listing of the communications region, TSSA.

AMFTM - (R) Maximum core fence maintenance time.

LNSF - (I) Number of swap files.

TASWT - (R) Minimum memory allocation wait time to cause further memory allocation and swap actions.

TAMIS - (R) Maximum high priority service program size.

TALPP - (R) Large program penalty factor.

TASWF - (R) Factor for program urgency calculation.

TASID - (R) Damper for urgent user size increases.

TLTLM - (R) Maximum time between line service calls.

TAMII - (I) Minimum memory size increase increment.

TAMMS - (I) Maximum TSS swap area size. (In the real system this is total TSS memory size.)

AMTQ - (R) Minimum core residency time before consideration for force swap.

TAGMI - (R) Minimum time between requests for core size increases.

TATMC - (R) Maximum time for core size change to stay pending.

TATMD - (R) Delay before scheduled size reduction is completed.

TAMAW - (R) Delay before informing user "NOT ENOUGH CORE TO RUN JOB".

TLNLM - (R) Minimum time between periodic line service functions.

TASMS - (I) Minimum TSS swap area size. (In the real system this is total TSS memory size.)

TAMRI - (R) Minimum time between size reduction considerations.

TAPMR - (R) Value of TAPMU above which memory reduction in not requested.

TASRI - (I) Memory size reduction amount.

TASCF - (R) Minimum interval between urgent user size increases.

ASD3I - (R) Minimum in interval between entries to SDP3 for scan of urgent users.

TCDEL - (R) Time slice for each subsystem dispatch.

4.3 Driving Data

The data under this category consists of the data collected from the actual system that is to be used to drive the model. All this data is obtained from the real systems using TSS accounting records and the programs mentioned in References 3 and 4.

INITCORE - (I) Initial TSS swap area size.

INTMEAN - (R) Mean interrupt interarrival times (cpu time/interrupt).

MNDELAY, MXDELAY - (R) Lower and upper limits for a uniform distribution of time delay after each interrupt.

DIOMEAN - (R) Mean disk I/O service time (real time).

OUTMEAN - (R) Mean duration of a keyboard output operation (real time).

USTIAT - Distribution of subsystem interarrival times (real time/subsystem).

NOKIN - (I) Distribution of keyboard inputs per subsystem.

NOKOUT - (I) Distribution of keyboard output per subsystem.

NODIO - (I) Distribution of disk I/O's per subsystem.

SWAPDUR - (R) Distribution of swap channel service time (real time).

CPUDUR - (R) Distribution of cpu time used by subsystems.

KIODUR - (R) Distribution of keyboard input durations (real time).

SIZEDIST - (I) Distribution of subsystem program sizes.

The input data must be terminated with the number 9999. The program uses this as a check to make sure the input data had the correct number of data elements.

Figure 2 shows a sample input data set.

```
10 10800000 0 0 0 0 0 0 0 0 10800001  0 3600000
20 98275 78985 75300 32388 52390
30 22154 24369 16815 69295 54224
40  20000  4          3000      36       4        .008     1000      500
50  7      58         7000      14000    50000    30000    150000    3000
60  20     300000     75        5        30000    1000     25
70  40     30       7        15       88.3    2200
80 0 7 .315 1192 .53 2741 .589 3370 .71 5233 .813 7323 .860 9414
90      .91 11500 .949 14640 .965 20910 .988 23000 .995 27100
100      .995 33450 .995 39720 1.0 47662 *
110   0 0 .528 0 .754 1 .845 2 .89 3 .899 4 .911 5 .93 6 .941 7
120       .951 8 .956 9 .958 10 .96 11 .97 12 .974 13 .977 14
130       .981 17 .984 18 .986 19 .988 20 .991 27 .993 29 .995 34
140       .998 38 1.0 150 *
150   0 0 .321 0 .684 1 .806 2 .871 3 .881 4 .892 5 .911 6 .923 7
160       .925 8 .927 9 .93 10 .934 11 .944 12 .956 13 .958 14 .963 17
170       .967 18 .97 20 .972 22 .979 24 .981 25 .984 29 .988 30
180       .991 35 .993 36 .995 58 .998 111 1.0 173 *
190   0 0 .340 0 .561 1 .581 2 .717 3 .778 4 .81 5 .834 6 .852 7
200       .876 8 .897 9 .902 10 .911 11 .923 12 .932 13 .944 14
210       .946 19 .948 20 .951 21 .953 25 .956 29 .958 30 .965 31
220       .967 33 .97 35 .979 40 .981 55 .984 62 .986 89 .988 99
230       .991 148 .993 552 .995 1101 .998 2023 1.0 4665 *
240   0 27.5 .015 85.3 .134 200.8 .218 247 .305 281.6 .534 327.8
250          .673 339.4 .829 431.8 .883 489.5 1.0 729 *
260   0 4 .131 12.3 .237 15.1 .365 22.5 .507 33.5 .648 40.9
270          .657 50.8 .789 97.6 .892 472.2 .913 800 .932 1400
280          1.0 39917 *
290 0 0  .001 2275  .023 5832 .234 6340 .386 11330 .509 13250
300      .507 17070 .515 18900 .535 22810 .54 24730 .738 28550
310      .753 32380 .798 35210 .851 38120 .865 41950 .915 45770
320      .930 53430 .951 55340 .963 61080 .965 76300 .971 78300
330      1.0 282381 *
340 0 1 .094 1 .153 2 .190 3 .271 4 .35 5 .364 6 .378 7 .364 8
350      .7 9 .717 10 .738 11 .817 12 .895 13 .91 14 .929 15
360      .934 16 .945 17 .952 18 .967 19 .959 20  .960 21
370      .971 22 .991 23 .991 24 .991 25 1.0 26 *
380  9999
390  DATA FOR MARCH 17  13.5 TO 14.00
```

Figure 2. Sample Input Data Set

5. Output

The normal simulation output consists of five categories as described below. The final section deals with output messages printed when the diagnostic switches are set on.

5.1 Input Echo

The first part of the output consists of an echo of the entire input data except for the distributions in Section 4.3.

5.2 Memory Statistics

The mean and standard deviation of the following TSS swap area statistics are printed.

1. Program sizes in core.
2. Hole sizes.
3. Total swap area size.
4. Total used core size.
5. Percentage core utilization.

The values in this section are accumulated automatically by Simscript every time their values are sampled in routines MMV and CORSAMP.

5.3 Overall Mean Rates

This section contains the mean rate of occurrence per hour of the following TSS events and processes.

1. Keyboard inputs.
2. Keyboard outputs.
3. Disk I/O's.
4. Swap outs (total).
5. Swap outs (due to keyboard I/O).
6. Swap outs (due to key output only).
7. Force swaps.
8. TSS swap area size increases.

9. TSS swap area size decreases.

10. Total subsystem CPU time.

11. Subsystem starts.

12. Subsystem terminations.

13. Dispatches of the CPU to subsystems.

14. Urgent users detected.

15. Entries to Processor Allocator.

16. Entries to Memory Allocator.

17. Allocator idles.

18. Interrupts.

19. Total core swapped.

These items are accumulated explicitly in the program and converted to hourly rates in routine OUTPUT.

5.4 Typical Subsystem Behavior

This section contains information on the behavior of a typical subsystem that executed during the simulation period. These statistics are collected at each subsystem termination as in the real system for the TSS accounting records. (See Ref. 3).

1. Subsystem interarrival time.

2. Keyboard input interarrival time.

3. Keyboard output interarrival time.

4. Disk I/O interarrival time.

5. Program size.

6. CPU time (sampled).

7. CPU time (used).

8. Response time (R19).

9. Response time (individual).

10. Time spent in each time state:

    (a) Non-useful core residency.

    (b) Swap.

    (c) Useful core residency.

    (d) Out of core.

    (e) Awaiting memory.

    (f) Awaiting memory after being force swapped.

11. Keyboard inputs.

12. Keyboard outputs.

13. Disk I/O's.

14. Force swaps.

15. Total swaps.

Items 1 through 6 are obtained from the samples of the input distributions and are useful for verifying that the input distributions were correct.

Items 6 and 7 differ because the former is the sampled time whereas the latter is computed only on those subsystems that terminate during the simulation period.

Item 8 is the response time as computed from the TSS accounting records in Reference 3. This response time is the total response time divided by the total number of keyboard inputs and outputs. The "individual" response time consists of samples of response time taken at every courtesy call for a keyboard operation.

5.5 System Queue Data

This section gives information on the average number of subsystems in each system queue as well as the average number of subsystems

```
BEGINNING ØF SIMULATIØN AT 0.


 STØPTIME DEVS DRS DQS DMS DSCS DUS
 10800000.000000 0 0 0 0 0 0



 DKS DSS DBEGIN DDUR INITIME
 0 0 10800001.000000 0. 3600000.000000

RANDØM NUMBER SEEDS
 57839
 84934
 62192
 72206
 58023
 34548
 34886
 88730
 75442
 78955


 AMFTM LNSF TASWT TAMIS TALPP TASWF TASID
 20000.000000 4 3000.000000 36 4 .008000 1000.000000



 TLTLM TAMII TAMMS AMTQ TAGMI TATMC TATMD
 500.000000 7 58 7000.000000 14000.000000 60000.000000 30000.000000



 TAMAW TLNLM TASMS TAMRI TAPMR TASRI TASCF
 150000.000000 3000.000000 20 300000.000000 75 5 30000.000000



 ASD3I TCDEL INITCØRE INTMEAN MNDELAY MXDELAY DIØMEAN
 1000.000000 25.000000 40 30.000000 7.000000 16.000000 88.299999



 ØUTMEAN
 2200.000000

SIMULATIØN TERMINATIØN AT 10800000.000


CØRE STATISTICS MEAN STANDARD DEVIATIØN
PRØGRAM SIZES 7.677 4.59893
HØLE SIZES 4.980 9.11604
SWAP AREA SIZE 48.785 8.08320
TØTAL USED CØRE 26.746 12.83447

PERCENT CØRE USED 54.265 24.29143
```

Figure 3. Sample Simulation Output

```
ØVERALL MEAN RATES (PER HØUR)
KEYBØARD ØUTPUTS 2338.00
KEYBØARD INPUTS 1665.50
DISK I/ØS 12188.50
SWAP ØUTS 1881.50
SWAP ØUTS (KEY I/Ø) 1827.00
SWAP ØUTS (KEY ØUTPUT) 551.50
FØRCE SWAPS 46.50
SIZE INCREASES 4.50
SIZE DECREASES 6.00
SUBSYSTEM CPU TIME 852245.906 MS.
SUBSYSTEM STARTS 760.50
SUBSYSTEM KILLS 754.50
SUBSYSTEM DISPATCHES 69012.50
URGENT USERS 47.00
ENTRIES TØ PAP 80558.00
ENTRIES TØ MAP 9302.00
ALLØCATØR IDLES 11545.50
INTERRUPTS 69003.00
TØTAL CØRE SWAPPED 13017.50 K


SUBSYSTEM STATISTICS MEAN STANDARD DEVIATIØN
SUBSYSTEM IAT 4737.04 6200.02
KEY INPUT IAT 178.42 1453.09
KEY ØUTPUT IAT 242.71 1790.60
DISK I/Ø IAT 62.39 881.83
PRØGRAM SIZE 8.14 5.03
CPU TIME (SAMPLED) 1146.56 5179.26
CPU TIME (USED) 1080.62 5006.96
RESPØNSE TIME (RIØ) 3238.88 22184.43
RESPØNSE TIME (INDIVIDUAL) 1740.16 13153.81
TIME IN STATE
 NØN-USEFUL CØRE 8159.79 31824.46
 SWAP 1607.56 4395.30
 USEFUL CØRE 5053.05 28368.11
 ØUT ØF CØRE 34956.78 150524.82
 AWAIT MEMØRY 347.57 1776.40
 AWAIT MEMØRY AFTER FS 30.91 336.93
NØ. ØF KEY INS 1.83 7.52
NØ. ØF KEY ØUTS 3.10 11.25
NØ. ØF DISK I/Ø'S 15.43 195.43
NØ. ØF FØRCE SWAPS .06 .43
NØ. ØF SWAPS 4.80 13.15


QUEUE LENGTHS MEAN STANDARD DEVIATIØN
PRØCESSØR QUEUE 1.11 1.17
MEMØRY QUEUE 11.23 2.91
DISK I/Ø QUEUE .40 .68
USERS SWAPPING .35 .80
USERS IN CØRE 3.48 1.78

URGENT USERS .02 .16
USERS WAITING FØR CØRE .48 .54
USERS ELIGIBLE FØR CPU .93 1.06
```

Figure 3 (Contd.). Sample Simulation Output

performing special activities.

1. Processor queue.
2. Memory queue.
3. Disk I/O queue.
4. Number of users swapping.
5. Number of users in core.
6. Number of urgent users.
7. Number of users waiting for core.
8. Number of users eligible for the CPU.

The queue lengths are maintained and accumulated by Simscript automatically. The other items are updated explicitly and accumulated automatically.

Figure 3, on the following page, shows a sample simulation output.

5.6 Diagnostic Output

This section describes the types of messages printed when each of the diagnostic switches described in section 4.1 are turned on.

1. DEVS - A message of the type

eeee AT ttttt.t

is printed whenever an event is entered. eeee is the event name and ttttt.t is the simulation time. For events that are associated with a subsystem (e.g. derails and courtesy calls), one of the phrases

BY nnnn

FOR nnnn

OF nnnn

TO nnnn

are postfixed to the message as appropriate. nnnn is the subsystem index number.

This switch also causes the printing of the message

START AT ttttt.t OF nnnn SIZE = cccc

CPU = sssss.s

in routine START. cccc and sssss.s are the program size and cpu time allocated to the new subsystem.

2. DRS - A message of the type

rrrr CALLED

is printed at the beginning of every routine execution. rrrr is the name of the routine. This message is not printed for routine START.

3. DQS - The MEMQ and CPUQ are printed whenever their members are changed or rearranged. The UST's in the queue are printed in the order in which they appear in the queue. For each UST the following line is printed.

nnnn b1 b2 b3 b4 b5 b6 b7 s

nnnn is the index number of the UST. b1 through b7 represent the flag word bits FL18, FL19, FL21, FL22, FL23, FL24 and FL34 respectively. s represents one of six states that the UST can be in. s takes on values from 0 through 5 and represents the following subsystem states:

0 Non-useful core residency.

1 Swap.

2 Useful core residency.

3 Out of core.

4 Awaiting memory.

5 Awaiting memory after force swap.

4. DMS - The memory map is printed whenever it is altered. For each entry in the map the following line is printed

nnnn pppp hhhh

where nnnn is the UST index number, pppp is its program size and hhhh is the size of the hole succeeding the program in core. The first entry in the core map, which is a dummy entry has an undefined value for nnnn and a program size of zero.

5. DSCS - A message is printed whenever TSS core size is changed. The two possible messages are:

SIZE INCREASED TO ssssK AT ttttt.t

SIZE DECREASED TO ssssK AT ttttt.t

where ssss is the new core size and ttttt.t is the time of the size change. In addition, whenever routine CORSAMP is called (at the start of every subsystem) the values of total, used and percent core used are printed as follows:

AT ttttt.t TOTCOR = sss USED = ttt = ppp.p% (MEAN = mmm.m%)

6. DUS - A message is printed whenever an urgent user is detected and a force swap is performed.

The urgent user message is:

nnnn FOUND URGENT FOR sssss.s MS. AT ttttt.t

where nnnn is the UST index number, sssss.s is the time elapsed since this UST was first discovered urgent, and ttttt.t is the current simulation time.

The force swap message is:

nnnn FORCE SWAPPED AT ttttt.t

7. DKS - A message is printed at every keyboard input and output derail and courtesy call. The derail messages are:

nnnn START INPUT AT tttt.t UNTIL ssss.s

nnnn START OUTPUT AT tttt.t UNTIL ssss.s

The courtesy call messages are:

nnnn FINISHED KIO AT ttt.t .

where nnnn and tttt.t are as before, and ssss.s is the time of the scheduled courtesy call for the I/O operation.

8. DSS - Messages are printed at the start and termination of every subsystem. The start message is identical to that for DEVS. The termination message is:

STOP AT tttt.t OF nnnn KIN = iii KOUT = jjj

TOT RESP = sss.s CPU ALLOC = uuuu.u USED = vvvv.v

where nnnn and tttt.t are as before, iii and jjj are the total keyboard inputs and outputs performed by the subsystem respectively, sss.s is the total response time accumulated. uuuu.u and vvvv.v are the CPU time allocated to the subsystem and total time used by it and must always be equal (See Sec. 8).

6. Using the Program

The program was tested and run on a Honeywell 6000 computer under the GCOS operating system. The use of this program is covered in two sections, compilation and execution.

6.1 Compilation

The program can be compiled by the CACI Simscript II.5 compiler for HIS 600/6000 comp ters (USAF release 9). Compilation requires approximately 61 words of storage and 7 minutes of processor time. The following deck would compile the program in the source file SIM and store the object deck in the file SIMOB.

```
1000   #S,U,J              ,8,16,32
1010   $        IDENT      BFCAUM01,MULLA J ,558102110001,UNIV. OF MICHIGAN
1020   $        PROGRAM    RLHS,ØN5,DECK
1030   $        LIMITS     13,61K,,10K
1040   $        PRMFL      H*,R,R,SMSCP2.5/CØMPILER
1050   $        FILE       *1,X1R,20L
1060   $        FILE       *2,X2R,20L
1070   $        FILE       B*,B1S,20L
1080   $        PRMFL      C*,R/W,S,BFCAUM01/SIMØB
1090   $        SELECTA    BFCAUM01/SIM
1100   $        ENDJØB
```

To compile the program from TSS the following underlined commands must be issued. It is assumed that the above lines are in the file COMP.

```
SYSTEM  ?CARDIN ØLD CØMP
READY
*RUN
  SNUMB # 1234T
*
```

To examine the output of the compilation from TSS the following commands must be entered.

```
SYSTEM ?JØUT  1234T
FUNCTIØN?SCAN 74
FØRM?DUMP
EDIT?YES
?PRINT /*** ERRØR/;*
?DØNE
FUNCTIØN?DIRECT ØNL
```

If no error comments are printed in response to the print command it indicates that the compilation was error-free.

6.2 Execution

To execute the program the following commands are required. Execution requires 25 K words of storage and approximately fifteen minutes of processor time for every hour simulated. It is assumed that the data to be used is in the file SIMDATA.

```
1000  #S,U,J           ,8,16,32
1010  $       IDENT    BFCAUM01,MULLA J ,558102110001,UNIV. ØF MICHIGAN
1020  $       LØWLØAD
1030  $       ØPTIØN FØRTRAN
1040  $       LIBRARY SL
1050  $       SELECT  BFCAUM01/SIMØB
1060  $       EXECUTE
1070  $       LIMITS  50,25K,-3K
1080  $       PRMFL   SL,R,S,SMSCP2.5/LIBRARY
1090  $       PRMFL   17,R,S,SMSCP2.5/ERRØRS
1100  $       FILE    B*,B1R
1110  $       DATA    I*
1120  $       SELECTA BFCAUM01/SIMDATA
1130  $       ENDJØB
```

To run the program from a terminal the following underlined commands must be entered. It is assumed that the above GCOS commands are in the file RUN.

```
SYSTEM  ?CARDIN ØLD RUN
READY
*RUN
  SNUMB # 5678T
*
```

To examine the output from the program on TSS the following commands must be issued.

```
SYSTEM ?JØUT 5678T
FUNCTIØN?PRINT 06
```

7. Detailed Program Description

The following is a detailed, line by line description of each routine and event in the program. The order is as in the program listing in Appendix 3.

7.1 PREAMBLE

8 - 11	Text substitutions for the compiler.
18 - 33	Random variable distribution entities. See Section 4.3.
38 - 49	Definition of UST entity and its attributes.
50	Definition of memory queue ordering on LSIZE attribute.
54 - 65	Definition of event routines.
66	Definition of CCQ set and its limited attributes.
70 - 71	Definition of memory map arrays.
72 - 91	Definition of global variables. See Appendix 1 for details.
95 - 134	Definitions of variables for automatic collection of statistics and global counters.

7.2 INIT

145 - 162	Read and echo input data from sections 4.1 and 4.2. (Diagnostic control and TSS parameters).
163	Read input values for distributions in Section 4.3. (Driving data).
164 - 167	Read check and stop if it is not 9999.
172	Reserve array locations for memory map vectors.
176 - 183	Initialize memory map. Link all blocks in a doubly linked list. Create dummy block as first one in map.
187 - 189	Schedule the arrival of the first subsystem and I/O interrupt.
192 - 195	Schedule entry to allocator termination of simulation, and event to test diagnostic switches.

7.3 MAIN

201	Call INIT to initialize program
202	Start simulation clock.

7.4 PDIAG

212 - 215	If the call is at the beginning of the diagnostic period set all intern switches to the values read in INIT.
218 - 220	At the end of the diagnostic period reset all switches to zero.

7.5 REINIT

226 - 233	Call Simscript generated routines to initialize counters for TALLY and ACCUMULATE variables.
235 - 238	Reset all global counters to zero.

7.6 CORSAMP

246 - 253 Scan through the memory map and add up the program sizes in USEDCORE and both program and holesizes in TOTCOR.

254 Calculate percent core used in PERUSED.

255 - 256 Print core size and percentage used for diagnostics.

7.7 MQPRINT

262 - 268 For each UST in the memory queue print the index number, flag word bits and state number.

7.8 CQPRINT

274 - 280 Do the same as MQPRINT for each UST in the processor queue.

7.9 STERM

286 Call OUTPUT to print simulation results and then terminate the simulation.

7.10 OUTPUT

291 - 292 Print time at which simulation was terminated.

296 - 304 Print memory utilization statistics.

308 - 331 Print the number of certain TSS events that occurred per hour during the simulation period.

335 - 364 Print statistics on an average subsystem the executed during the period.

368 - 380 Print statistics on queue lengths.

7.11 SNAP.R

389 - 393 Print next arrival time for each event in the program.

394 Print arrival time of derails for the UST that was being served last by the allocator.

397 - 398	Print arrival times of all courtesy calls in the CCQ.
401	Call OUTPUT to print simulation results up to this point.

7.12 ALLOCI

409 - 410	Call LINSV or MAP if their respective flags are set.
413 - 414	Increment count of entries to allocator and set allocator flag.
41[illegible] - 417	Scan CPUQ for jobs that are not doing I/O and not swapping.
419	See if job found is scheduled for force swap. If so call SDP7.
420	If not call RETSSX to dispatch cpu to job.
421	Reset the UST's new subsystem bit.
426 - 428	If no jobs eligible for cpu, reset allocator flag and go to LINSV.

7.13 MAP

436	Increment count of entries to MAP.
437	Reset map flag.
440	If special memory action flag is [illegible] call SPMACT.
441	Set time of last entry into MAP
443	If no urgent user is waiting go to MP.2A.
444	If urgent user can fit in current [illegible] fence go to MAP.3.
445 - 447	If he has been urgent for more than [illegible]TM ms. reject him and clear the core fence.
451 - 452	Scan MEMQ for jobs that are not doing keyboard I/O, not in core and not swapping.
454	If such a job was not force swapped to go to MAP.4 and try to swap him in immediately.

455	Otherwise remember the first job we find eligible for swap in.
459 - 460	If no swap candidate was found and no special memory action is waiting return to ALLOCI.
461 - 463	If special memory action waiting, call swap decision processor.
467	If swap candidate was found call MBA to allocate storage.
470 - 473	If allocation was unsuccessful call swap decision processor.
476	If successful, call SWIN to swap him in.
477 - 478	Return to MAP to allocate storage for the next program.
481 - 484	If urgent user got core, destroy the core fence.
485 - 487	If he is in the MEMQ and still needs core call MBA to allocate it. Otherwise go to MAP.3 to look for another swap candidate.

7.14 SDP

496 - 497	If more swaps are in progress than there are swap files, go back to ALLOCI.
498	Reset the pointer to job to be swapped.
501 - 502	Scan the MEMQ for jobs that are doing keyboard I/O and are in core and not swapping.
504 - 505	If the size of such a job exceeds the required size set AMN1 to that UST.
509 - 510	If no swap candidate was found call SDP3.
513 - 515	Call SWOUT to swap out job found.
518	Repeat SDP for more jobs to swap out.

7.15 SDP3

526 - 529	Enter only if last entry was at least ASD31 ms. ago or five unsuccessful attempts were made previously.
530 - 535	Reset counter, entry time, number of urgent users found, memory needed, largest urgent user wait time.
539 - 543	Find user that needs memory and has been waiting at least TASWT ms.
547 - 550	Calculate job wait factor depending on size of program.
551 - 552	See if wait factor calculated is less than time waited. If so he is eligible to be urgent.
555 - 560	If he was force swapped, reset the force swap bit and change his state from "waiting memory after force swap" to "waiting memory", but do not consider him urgent.
563 - 564	If he was found urgent, increment urgent user counts.
566 - 567	If USWITCH is set print urgent user message.
568 - 569	If he is the longest waiting urgent user, set ITALUT to his UST.
572 - 573	Record number of urgent users detected in this pass through SDP3. Call SDP4.

7.16 SDP4

581 - 582	If no urgent user was found call SDP6.
583 - 589	Call SDP5 if (i) longest wait time is less than TASID ms., (ii) it has been less than TASCF ms. since the last size change, or (iii) a size change request is already scheduled.
592 - 298	Calculate a new (larger) size for TSS swap area.
599	Increment size increase requests.

600 Set special memory action flag.

601 Call SDP5 to set up core fence.

7.17 SDP5

610 If core fence is already up call SDP6.

611 - 613 If urgent user size is greater than TSS swap area size call MBA3 to increase size.

614 - 615 Set core fence size (2AURWT) needed to LSIZE of urgent user.

616 Set time fence was established.

618 Call SDP6 to force swap jobs.

7.18 SDP6

627 If special memory action waiting go directly to scan for force swap jobs.

628 - 632 If an urgent user was found and swap files are not full begin scan for force swaps. Otherwise return to ALLOCI.

633 - 635 Scan CPUQ for a job that has been in core more than AMTQ ms.

636 If the job is not new and is not already scheduled for a force swap, set the force swap bit.

637 - 638 If the job is doing I/O return to ALLOCI.

639 - 640 Otherwise call SDP7 to force swap it immediately.

643 Return to ALLOCI.

7.19 SDP7

652 - 659 Call SWOUT to swap job out. Increment force swap counters for UST and system. Move UST from CPUQ to MEMQ. Print diagnostic messages for QSWITCH and USWITCH.

660	Return to ALLOCI.

7.20 SPMACT

670 - 672	If no size change is scheduled, clear flag and return.
673	If size decrease is scheduled go to SPM.3.
676	If size has been increased less than TAGMI ms. ago, ignore this request.
678 - 679	Print diagnostics for increase.
683	If new size requested is current size or less than 3K, ignore it.
684 - 688	If there is no hole at the tail end of core large enough for the reduction, or the request has been pending longer than TATMC ms. or less than TATMD ms. go to SPM.5 to see if request should be ignored.
690 - 691	Print diagnostics for decrease.
695	Update memory map entry.
696 - 697	Clear size request and request flag.
699	Set time of last change or change attempt.
703 - 706	Ignore request if time since last request is greater than TATMC.

7.21 MBA

714	Clear success return flag.
715	Increment count of entries.
719 - 726	Check memory map for a hole big enough to satisfy request.
729 - 731	If search was unsuccessful and program will not fit in TSS size, call MBA3.

734 - 737 If allocated job was the urgent user, clear the core fence.

738 - 740 If he is not the urgent user, make sure he does not use up the core fence set up for the urgent user.

744 - 759 Build and link new entry into the memory map.

760 Set flag to indicate that allocation was successful.

761 Call MMV to verify memory map is still good.

7.22 MBA3

770 - 772 If urgent user has waited more than TAMAW ms. for core, return.

773 - 775 If a size increase is already scheduled, return.

776 Set new request to size of program.

777 Set the special memory action flag.

778 - 780 If urgent user needs more than TAMII plus the current TSS swap area size, return.

781 - 783 Otherwise set the new size request to TAMII more than the current size but not more than TAMMS.

7.23 MBD

793 - 797 Scan memory map for the job with the same pointer as the one to be deallocated.

800 - 807 Remove the entry from the core map and relink preceding and succeeding entries.

808 If the deallocated program was larger then 2K, set the MAP flag.

809 Call MMV to verify the memory map.

7.24 MMV

823 - 832	Scan every entry in the memory map and add up each program and hole size in TOTCOR.
825 - 827	Print diagnostics when MSWITCH is set.
833	Sample value of CORSIZE for statistics.
837 - 839	Compare TOTCOR with the current TSS swap area size. If they are not equal print an error message and stop.
844 - 848	Scan the MEMQ for jobs waiting for core (i.e., not doing I/O, not swapping, not in core and not new subsystem).
849 - 853	Scan the CPUQ for jobs eligible for the cpu (i.e., not doing I/O, not swapping, not scheduled for a force swap and in core).

7.25 SWOUT

861	Clear "in core" bit.
862	Set "swapping out" bit.
863	Increment swap count.
864 - 865	If keyboard I/O is in progress, set "roadblocked" bit, and clear "scheduled for force swap" bit.
866	Call SWPLD to swap out.

7.26 SWIN

875	Set "swapping in" bit.
876	Call SWPLD to perform swap in.
877	Increment count of jobs in core.

7.27 SWPLD

883	DIR specifies direction of the swap to be performed. (0 = in, 1 = out.)

886	Increment subsystem's swap count.
887	Change subsystem state to "swapping".
888	Increment count of jobs swapping.
890 - 891	Schedule swap in courtesy call. File the courtesy call in CCQ.
892	Schedule swap out courtesy call.
893	Add program size to total core swapped out.
894	File the courtesy call in CCQ.

7.28 SSFINI

902	Increment count of subsystem terminations.
903 - 904	Decrement count of jobs in core. Change subsystem state to "out of core".
905	Call MBD to release core.
908 - 911	Remove job from whichever queue it was in and print queue diagnostics if necessary.
914 - 920	If an output courtesy call is still pending for this subsystem, cancel it.
921	Call SACT to take subsystem accounting and destroy UST entity.

7.29 BUFDMP

930	For output operation, bypass this routine and call KIOSRT.
931 - 933	For input, move the UST from the CPUQ to the MEMQ.
934	Change subsystem state to "non-useful core residency".
935	Set "I/O roadblocked" bit.
937	Print queues for diagnostic purposes.

938	Call KIOSRT to start I/O operation.

7.30 KIOSRT

950 - 954	If keyboard I/O is already in progress (FK19 = 1), it must be an output since the subsystem was executing. In this case, cancel the output courtesy call.
955	Set "I/O in progress" bit.
958	If operation is an output:
959	Increment subsystem's count of outputs.
960 - 965	If no output is in progress collect response time value since the last key I/O courtesy call.
966	Set LTIN to the arrival time of the courtesy call.
967 - 969	Schedule the output courtesy call and file it in the CCQ.
970 - 971	Print diagnostic message if KSWITCH is set.
975	If operation is an input: Increment subsystem's count of inputs.
976 - 981	If no previous output is in progress, collect a response time sample.
982 - 986	Schedule input courtesy call and print diagnostic message if KSWITCH is set.

7.31 START

992	Increment count of subsystem arrivals.
994	Call CORSAMP to sample values for memory map statistics.
997	Create a new UST entity.
998 - 999	Sample, assign and tally values for program size and cpu time allocated.

100 - 1006	Sample values for the number of disk I/O's, keyboard inputs and outputs for this subsystem, and calculate the mean interarrival time between each. If zero operations are sampled the interarrival time is set to RINF.C (Simscript constant for the largest possible real number).
1007	Set time until next disk I/O, and key input and outputs as half the interarrival time. The I/O operations are equally spaced in cpu time throughout the subsystem execution period.
1008	Set "new subsystem" bit.
1009	Assign sequential index number.
1010	Initialize LTIN, LTMWT.
1011	File the UST in the memory queue.
1012 - 1013	Print message for diagnostics.
1014	Change subsystem state to "awaiting memory".
1015	Print MEMQ if QSWITCH is set.

7.32 RETSSX

1026 - 1027	Set NXTCC to the arrival time of the next courtesy call.
1028	Set NEXT to the smallest of the following times:

Time until the next disk I/O.

Time until the next key input.

Time until the next key output.

Cpu time remaining for the subsystem.

Time until the next interrupt.

Subsystem quantum (TCDEL).

Time until next courtesy call.

1029	Initialize DISPT to the time of dispatch.
1030 - 1034	Reduce each of the above times by the minimum value.
1038 - 1042	Schedule the derail which was to occur next. In case TCDEL, NXTINT or NXTCC was the least time, schedule an EXENTR to re-enter the allocator via an interrupt.

7.33 SACT

1052 - 1062	Sample each of the following values from the UST entity and assign them to their respective global tallied variables: Time spent in each of the six subsystem states. Keyboard inputs. Keyboard outputs. Disk I/O's. Force swaps. Swaps.
1065 - 1067	Calculate response time by dividing total response time accumulated by the total number of keyboard I/O's. If no key I/O's were done, response time is total subsystem duration.
1068	Sample subsystem cpu time allocated.
1069 - 1072	Print message if SSSWITCH is set.

7.34 ATCHG

1080 - 1086	Add current time minus time of previous call to ATCHG for this subsystem to the accumulated time for the previous state (specified by LTCW).
1087	Set LTCW to new state (N).
1088	Update time of last state change.

7.35 EXENTR

1097 - 1098	Move the interrupted UST to the last position in the CPUQ.
1099	Print CPUQ for diagnostic aid.
1100	Call EXEACT to take accounting.
1101	Re-enter the allocator via ALLOCI.
1104	Sample the arrival time of the next interrupt.

7.36 EXEACT

1115	Increment count of interrupts.
1116 - 1118	Add cpu time used by subsystem to totals for subsystem and overall system.
1119	If MAP was last called more than ASD3I ms. ago, set MAP flag.
1120	If LINSV was last called more than TLTLM ms. ago, set LINSV flag.

7.37 KONDRL

1128	Set input operation flag.
1129	Call EXEACT to take accounting.
1130	Increment global key input count.
1131	Call BUFDMP to process input.
1134	Reset time until next input to mean input arrival time.
1135	Return to ALLOCI.

7.38 KOTDRL

1143	Set output operation flag.
1144	Call EXEACT to take accounting.
1145	Increment global key output count.

1146	Call BUFDMP to process output.
1149	Reset time until next output to mean output arrival time.
1150	Return to ALLOCI.

7.39 DRLDIO

1158	Set "disk I/O in operation" bit.
1159	Call EXEACT to take accounting.
1160 - 1161	Increment UST's and system's count of disk I/O operations.
1162	Reset time until next disk I/O to mean disk I/O arrival time.
1166 - 1168	If the DIOQ is empty, schedule the courtesy call for this UST and file it in the CCQ.
1169	File the UST at the end of the DIOQ.
1170	Return to ALLOCI.

7.40 DRLRET

1178	Call EXEACT to take accounting.
1181	Check to see that the subsystem utilized all the cpu time allocated.
1182 - 1183	If not, print an error message.
1184	Call SSFINI to terminate the subsystem.
1185	Return to ALLOCI.

7.41 KIOCC

1191	Save UST pointer in TUST and set UST to the subsystem for which KIOCC was scheduled.
1192 - 1193	Print message if KSWITCH set.
1196	Set time of courtesy call in LTIN.
1197	Reset "data in transmission" bit.

1200	If operation was an input:
1202	Reset "roadblocked" bit.
1203 - 1207	If program is not swapping out and not in core, move the UST from the MEMQ to the CPUQ. Change state to "useful core residency".
1208	Print MEMQ and CPUQ for diagnostics.
1211	If program is not in core, set MAP flag, and change state to "awaiting memory".
1213	Restore UST pointer and return.

7.42 DIOCC

1220	Save UST pointer as in KIOCC.
1223	Remove the UST from the top of the DIOQ. He must be the one whose courtesy call is being serviced.
1224	Clear "disk I/O in progress" bit.
1227 - 1228	Move the UST to the top of the CPUQ.
1229	Print CPUQ for diagnostics.
1230	Restore UST pointer.
1234 - 1237	If there are more UST's in the DIOQ, schedule the first one.

7.43 1ALLCC

1244	Save the UST pointer as in KIOCC.
1247	Decrement count of users in core.
1248	Clear "swapping out" bit.
1249	Set MAP work flag on.
1251 - 1253	If UST was not doing key I/O and was force swapped change state to "awaiting memory after force swap".

1255	If UST was not force swapped and not doing key I/O change state to "awaiting memory".
1258	If UST was doing key I/O change state to "out of core".
1259	Call MBD to deallocate core.
1260	Decrement count of jobs swapping.
1261 - 1262	Restore UST pointer. Remove the courtesy call from the CCQ and destroy it.

7.44 2ALLCC

1268	Save UST pointer as in KIOCC.
1271	Set "program in core" bit.
1272 - 1273	Clear "swapping in" and "scheduled for force swap" bits.
1274 - 1275	Move the UST from the MEMQ to the top of the CPUQ.
1276	Change state to "useful core residency".
1277	Print MEMQ and CPUQ for diagnostics.
1278	Decrement count of users swapping.
1279 - 1280	Restore UST pointer, remove this 2ALLCC call from the CCQ and destroy it.

7.45 LINSV

1288	Clear LINSV flag.
1289	Clear "no users" switch.
1290	Reset time of last call to LINSV.
1292	Test if at least TLNLM ms. have passed before last entering the next section. If not, bypass the next section and go to MSRK.
1293	Update TLOLD, time of entry to this section.
1295	Test to see if both CPUQ and MEMQ are empty. If so, set MSR240 indicating no users.

1296	If the current core size is not at the minimum go to MSRKS to try to reduce it.
1297	If core is at its minimum, go to MSR300 to relinquish.
1301	Clear "no users" flag.
1302	If less than TAMRI ms. passed since the last time through the next section, skip to MSR300 to scan for new subsystems.
1306 - 1309	Calculate percentage core being used and average it with the previous value.
1312 - 1318	Find the largest program in the system.
1319	If no users, skip to MSR250.
1320 - 1325	Skip memory reduction if either of the following is true: (i) Time since last here is less than TAMRI ms. (ii) An urgent user was found since the last time through here. (iii) The last size change was less than TATMD ms. ago. (iv) A program has been awaiting memory since the last time through here. (v) The percent core utilization (TAPMU) is greater than TAPMR.
1326 - 1327	If current size minus TASRI is less than the minimum size, set change request to minimum. Otherwise set request to reduction by TASRI.
1328	If a change request is already scheduled, ignore this one.

1329	If new size is less than largest program found, ignore it.
1330	If time since last change is less than TASCT, ignore this request.
1331	Set special memory action flag.
1332	Set TAHOL to the new requested core size.
1333	Increment count of reduction requests.
1339	If the arrival time of the next UST is past, call START to create it.
1340 - 1341	Sample time of next UST arrival.
1342	Loop back to see if next arrival is also past.
1343	If either Allocator or MAP flags are set, return to ALLOCI.
1348	Increment count of idles.
1349	If MAP was last called more than ASD3I ms. ago, set MPWF.
1350 - 1351	Set time of next courtesy call.
1352 - 1353	Schedule an entry into the allocator at the arrival of the next UST, courtesy call or TAGMI ms., whichever is smallest.

8. Error Messages

This section describes only those error messages printed by the program. For Simscript generated error messages see Reference 5.

1. After reading the input data the program reads a check value and compares it with a preset value (see Section 7.2). If this value is not correct the program terminates with the following error message.

ERROR - INPUT FORMAT INCORRECT

2. When the memory buffer allocator (MBA) is called to find memory for a certain job, it checks to see if the program is already in core. If such a call is made the error is considered fatal and the following message is printed.

ERROR - MBA CALLED FOR UST ALREADY IN MEMORY

3. The memory map arrays (see section 3) are all dimensioned 50. If more than fifty jobs are ever allocated memory, the MBA process prints the following message and terminates the program. In this case the dimension figure in the INIT routine must be increased.

ERROR - NO MORE AVAILABLE BLOCKS FOR MEMORY MAP ARRAY

4. The memory map verification routine (MMV) prints a fatal message if the map does not verify. This occurs if the sum of all programs and holes in the map does not equal the total core size. The message printed is:

ERROR - MEMORY MAP DOES NOT VERIFY

5. The only non fatal error occurs when a program terminates without using up all the cpu time allocated to it. In such a case the return derail (DRLRET) prints the following message:

ERROR - nnnn ALLOCATED

CPU = tttt.t USED = ssss.s

where nnnn is the UST index number and tttt.t and ssss.s are the allocated and used cpu times respectively.

APPENDIX 1

Variable Description

The following is an alphabetically ordered list of variable names used in the program and their functions. When a name is followed by (M/S) it means that its mean and standard deviation are automatically collected by Simscript in the variables Mname and Sname respectively.

For certain TSS system constants, their values at the time this program was debugged are given in parentheses. Most variable and parameter names are the same as those in TSS and can be found described in Reference 2.

ALOCI - Number of entries to the allocator.

ALUTM - Time at which urgent user was last detected.

AMAP - Number of entries to MAP.

AMAP2 - Number of times the core fence limit was exceeded.

AMBA - Number of entries to MBA.

AMBA4 - Number of times a hole fit was detected by MBA.

AMBA5 - Number of successful allocations after hole fit detected.

AMFTM - Maximum force fence maintenance time (20000 ms.).

AMN1 - Pointer to UST for SDP.

AMN2 - Size of memory requested from SDP.

AMTQ - Minimum core residency time (7000 ms.).

APAP1 - Number of allocator idles.

APAP2 - Number of times subsystem selected for processor allocation.

ASDP - Number of entries to SDP because program awaiting memory.

ASDP7 - Number of force swaps.

ASD3C - Counter to override timer for entry to SDP3.

ASD3I - Minimum time between entries to SDP3 (1000 ms.).

ASD3T - Time of last entry to SDP3.

AVAIL - Pointer to the list of available blocks in the memory map arrays.

CCQ - Queue of pending courtesy calls. Ordered FIFO.

CHANGE - Temporary variable to hold core size change in LINSV.

CHECK - Check value (9999) to verify that data was read correctly in INIT.

CHKCPU - Allocated cpu time attribute of UST. Used to check used against allocated cpu time.

CORSIZE - Global sample of TACOR in MMV (M/S).

CPUDUR - Input distribution of cpu time per subsystem.

CPUQ - Processor queue.

DBEGIN - Time at which diagnostic printing is to start.

DDUR - Duration of diagnostic period.

DEVS - Diagnostic switch for event trace.

DIOIAT - UST attribute, disk I/O interarrival time.

DIOMEAN - Input mean of disk I/O service time for exponential distribution.

DIOQ - Disk I/O queue. Ordered FIFO.

DIR - Argument for SWPLD routine to specify direction of swap. 1 = out, 0 = in.

DISKIO - Global count of disk I/O's.

DISPT - Time at which last subsystem was dispatched.

DKS - Diagnostic switch for key I/O trace.

DMS - Diagnostic switch for memory activity trace.

DQS - Diagnostic switch for queue activity trace.

DRS - Diagnostic switch for routine trace.

DSCS - Diagnostic switch for core size and precentage core utilization trace.

DSS - Diagnostic switch for subsystem start and termination trace.

DUS - Diagnostic switch for urgent user and force swap trace.

DUST - Temporary location for UST in DIOCC.

ELIGCPU - Number of UST's eligible for the processor (M/S).

ES - Temporary index.

EVDIAG - Short for "FOR ES=1 TO EVSWITCH".

EVSWITCH - See DEVS.

EXT - Variable for accumulating time used to execute TSS code. The values for EXT were calculated from the actual TSS listings with the assumption that the average instruction execution takes approximately 2.2 microseconds.

FK19 - UST attribute bit, "data in transmission."

FL18 - UST attribute bit, "disk I/O in progress."

FL19 - UST attribute bit, "keyboard I/O in progress."

FL21 - UST attribute bit, "swap out in progress."

FL22 - UST attribute bit, "program in core."

FL23 - UST attribute bit, "swap in in progress."

FL24 - UST attribute bit, "new subsystem."

FL34 - UST attribute bit, "force swap scheduled."

HEAD - Pointer to the head entry in the memory map.

HOLSIZE - Samples of hole sizes in memory map (M/S).

I - Temporary index.

IDPTR - Memory map array to hold UST pointer.

INCORE - Number of UST's in core (M/S).

INDDRL - Flag set by keyboard I/O derails. 1 = input, 0 = output.

INITCORE - Input data value of initial TSS swap area size.

INITIME - Input data value of time at which statistic counters are to be reinitialized via event REINIT.

INTMEAN - Input data value of mean interrupt interarrival time for exponential distribution.

J - Temporary index.

JOBNO - UST attribute, subsystem index number.

KEYIN - Global count of keyboard inputs.

KEYOUT - Global count of keyboard outputs.

KIIAT - UST attribute, keyboard input interarrival time.

KILL - UST attribute, remaining cpu time until termination.

KIODUR - Input data distribution, keyboard input duration.

KOIAT - UST attribute, keyboard output interarrival time.

KOSWAP - Global count of swaps due to keyboard output activity after subsystem termination.

KSWITCH - See DKS.

KUST - Temporary location for UST in KIOCC.

LNSF - Maximum number of swap files (4).

LSFLG - Line service flag.

LSIZE - UST attribute, program size.

LSPTS - UST attribute, subsystem processor time used.

LSTIO - UST attribute, disk I/O count.

LTCW - UST attribute, current time state.

LTC21 - UST attribute, keyboard input count.

LTC22 - UST attribute, keyboard output count.

LTC31 - UST attribute, force swap count.

LTC32 - UST attribute, swap count.

LTIN - UST attribute, response timer.

LTMRS - UST attribute, accumulated response time.

LTMWT - UST attribute, working timer.

LTMO - UST attribute, non useful core residency time.

LTMOSS - Global sample of LTMO (M/S).

LTM1 - UST attribute, swap time.

LTM1SS - Global sample of LTM1 (M/S).

LTM2 - UST attribute, useful core time.

LTM2SS - Global sample of LTM2 (M/S).

LTM3 - UST attribute out of core time.

LTM3SS - Global sample or LTM3 (M/S).

LTM4 - UST attribute, waiting for core time.

LTM4SS - Global sample of LTM4(M/S).

LTM5 - UST attribute, waiting for memory after force swap time.

LTM5SS - Global sample of LTM5 (M/S).

MAPTM - Time of last entry to MAP.

MEMALOCOK - Return value from memory buffer allocator (MBA).
0 = unsuccessful, 1 = successful.

MEMQ - Memory queue, ordered by increasing LSIZE.

MNDELAY - Input data value, minimum of uniform distribution for interrupt delay.

MPACT - Special memory action flag.

MPWF - MAP work flag.

MS. - Short for "UNITS".

MSR240 - "No users" flag in LINSV.

MSWITCH - See DMS.

MXDELAY - Input data value, maximum of uniform distribution for interrupt delay.

N - Argument for ATCHG, new time state of UST.

NEXT - Cpu time interval allocated to subsystem in RETSSX.

NODIO - UST attribute, number of disk I/O's.

NOKIN - UST attribute, number of keybaord inputs.

NOKOUT - UST attribute, number of keyboard outputs.

NXTCC - Time until next courtesy call.

NXTDIO - UST attribute, cpu time until next disk I/O.

NXTINT - Cpu time until next interrupt.

NXTKIN - UST attribute, cpu time until next keyboard input.

NXTKOUT - UST attribute, cpu time until next keyboard output.

NXTUST - Real time until next subsystem arrival.

OUTCC - UST attribute, pointer to pending output courtesy call.

OUTMEAN - Input data value, mean output duration.

PERUSED - Percent core utilization (M/S).

PRED - Memory map array of backward pointers.

PROGSIZE - Samples of program sizes in memory map (M/S).

QSWITCH - See DQS.

RDIAG - Short for "FOR RS=1 TO RSWITCH PRINT 1 LINE THUS".

RESPT - Global sample of response time calculated for individual key I/O operations (M/S).

RS - Temporary index.

RSWITCH - See DRS.

R6 - Temporary UST pointer in MAP.

SCSWITCH - See DSCS.

SFUSE - Number of UST's swapping (M/S).

SHOLE - Memory map array to hold hole values.

SHOW - Short for "PRINT 1 LINE WITH TIME.V".

SIZEDIST - Input data distribution of program sizes.

SIZEINCR - Number of TSS swap area size increases.

SIZERED - Number of TSS swap area size reductions.

SJOB - Memory map array to hold program sizes.

SPMFR - Time of last size increase request.

SSCPU - Global sample of .LSPTS (M/S).

SSDIO - Global sample of LSTIO (M/S).

SSFSWAP - Global sample of LTC31 (M/S).

SSKIN - Global sample of LTC21 (M/S).

SSKOUT - Global sample of LTC22 (M/S).

SSRESP - Global sample of response time calculated by dividing the total response time by total keyboard I/O's (M/S).

SSSWAPO - Global sample of LTC32 (M/S).

SSSWITCH - See DSS.

STOPTIME - Input data value, simulation duration.

SUC - Memory map array for forward pointers.

SUST1 - Temporary variable to hold UST pointer in 1ALLCC.

SUST2 - Same as SUST1 for 2ALLCC.

SWAPDUR - Input data distribution, swap channel service time.

T - Temporary variable.

TAAUG - Number of urgent users detected.

TACOR - TSS swap area size.

TAGMI - Minimum time between size increases (14000 ms.).

TAGPT - Cumulative cpu time used by TSS since last relinquish to GCOS.

TAGTC - Number of interrupts.

TAGTU - Sum of subsystem cpu time.

TAHOL - New requested TSS swap area size.

TAIL - Pointer to last entry in memory map.

TALCT - Time of last size change attempt.

TALPP - Large program penalty multiplier for wait factor calculation in SDP3 (4).

TALPS - Largest program detected in LINSV size reduction logic.

TALUT - Longest urgent user wait time.

TAMAW - Maximum wait time for core before rejection of program (150000 ms.).

TAMII - Minimum memory size increment (7K).

TAMIS - Maximum size of programs considered under high priority (36K).

TAMMS - Maximum TSS swap area size (58K).

TAMRI - Minimum time between size reduction attempts (300000 ms.).

TAPMR - Minimum value of TAPMU required to prevent size reduction (75%).

TAPMU - Moving average percent core utilization.

TASCF - Minimum time between size increase requests due to urgent users (30000 ms.).

TASID - Damper for urgent user size increases (1000 ms.).

TASIO - Count of swap outs due to key I/O.

TASMS - Minimum TSS swap area size (20K).

TASRI - Memory size reduction amount (5K).

TASRT - Time of last entry to size reduction logic.

TASWF - Divisor to convert size into time units for wait factor calculations in SDP3 (.008).

TASWT - Minimum wait time before swap decision logic is invoked (3000 ms.).

TATMC - Maximum time allowed for size change to occur (60000 ms.).

TATMD - Delay before scheduled size reduction request is completed (30000 ms.).

TATMN - Total program size belonging to urgent users.

TAURG - Current number of urgent users (M/S).

TAUSE - Temporary variable to add up size being used.

TCDEL - Subsystem cpu time slice (25 ms.).

TEMP - Temporary variable.

TKILL - Number of subsystem terminations.

TLFLG - Allocator work flag.

TLLST - Time of last entry to LINSV.

TLNAA - Count of TSS idles.

TLNLM - Interval between line service functions (3000 ms.).

TLOLD - Time of last entry to idle status check in LINSV.

TLTLM - Minimum time between scan for new UST's in LINSV (500 ms.).

TOTCOR - Temporary variable to accumulate TSS core size in MMV.

TSIRC - Number of times size increase was requested because of waiting program.

TSRRC - Number of size reduction requests.

TSTRT - Number of subsystem starts.

TSWAP - Number of program swaps.

TSWPK - Total core swapped.

TUST - Temporary UST pointer storage.

URGT - Urgent user wait time.

USEDSIZE - Amount of TACOR being used by programs (M/S).

UST - Global pointer to UST currently being served by the allocator.

USTIAT - Input data distribution, UST interarrival times.

USWITCH - See DUS.

VCPUDUR - Sampled values of CPUDUR (M/S).

VDIOIAT - Sampled values of DIOIAT (M/S).

VKIIAT - Sampled values of KIIAT (M/S).

VKOIAT - Sampled values of KOIAT (M/S).

VSIZE - Sampled values of SIZEDIST (M/S).

VUSTIAT - Sampled values of USTIAT (M/S).

WAITCOR - Current number of users waiting for core (M/S).

WAKET - Time to reenter TSS after relinquish to GCOS.

WTFAC - Wait factor calculated for subsystem.

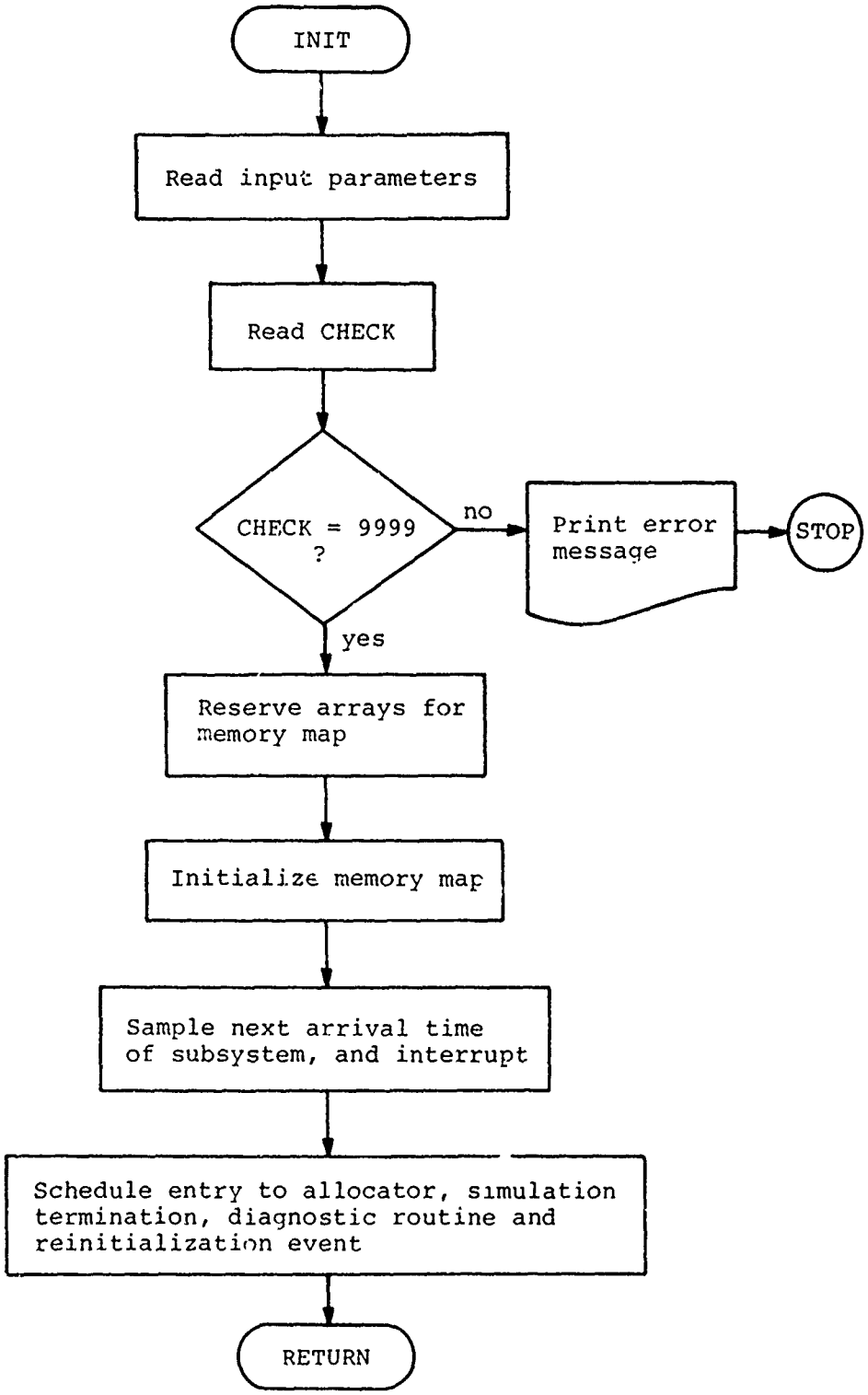
1AURWT - UST pointer to urgent user for whom core fence is established.

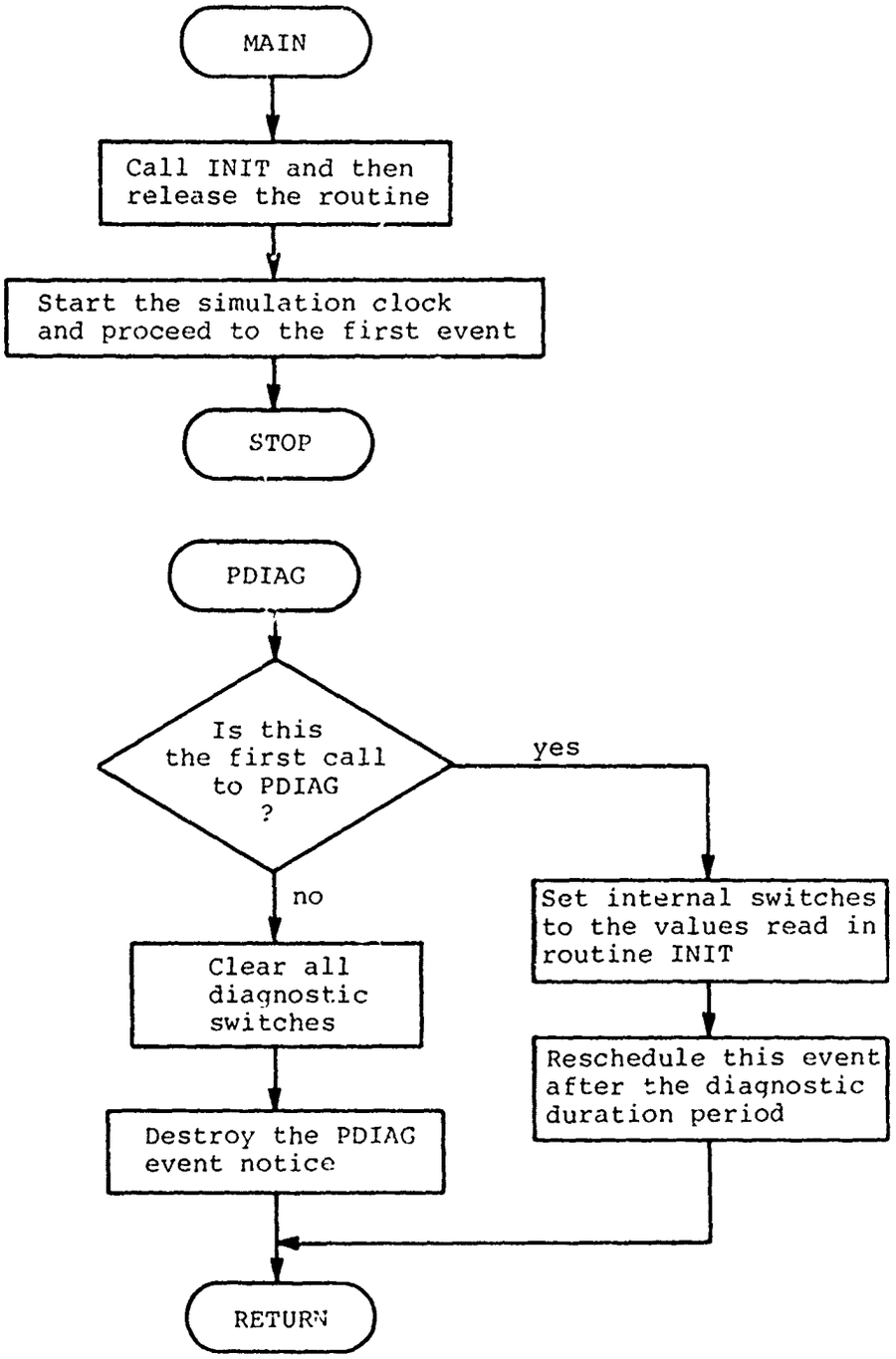
1TALUT - Pointer to UST that has been urgent the longest.

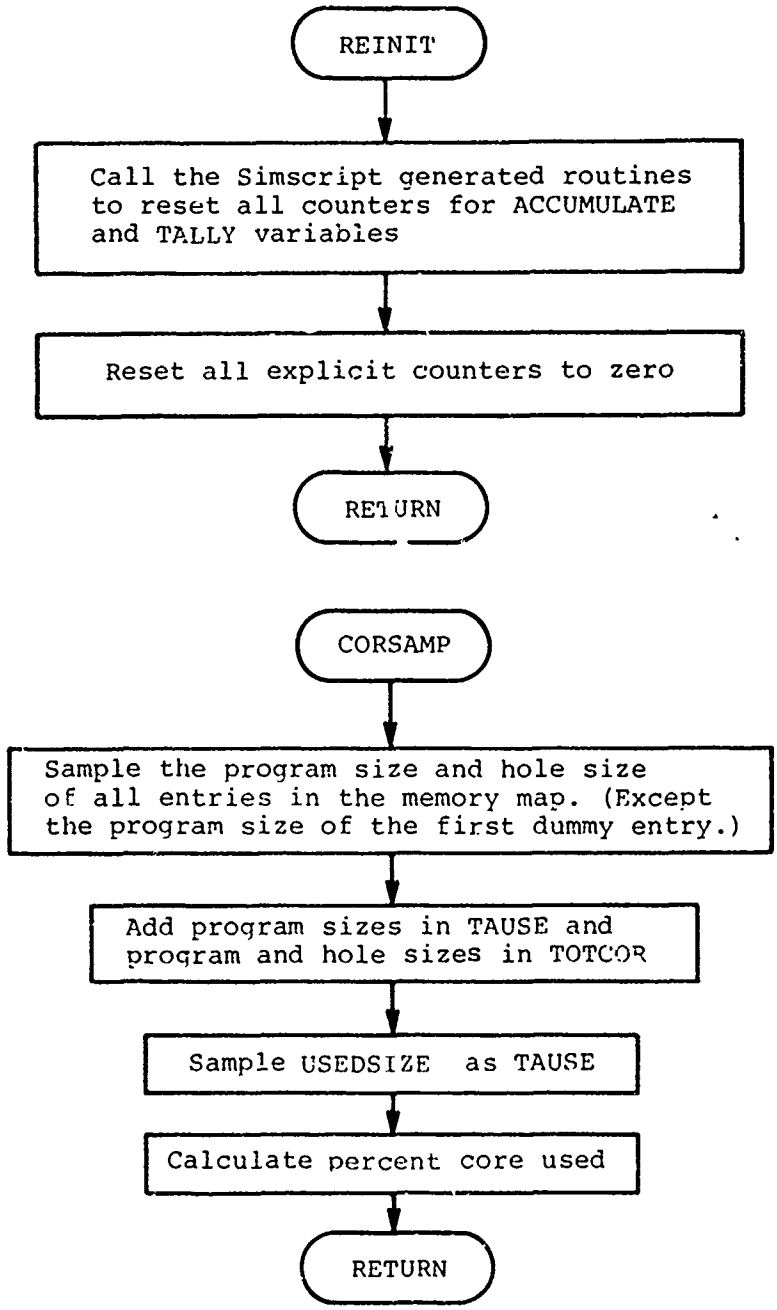
2AURWT - Program size of UST for which core fence is established.

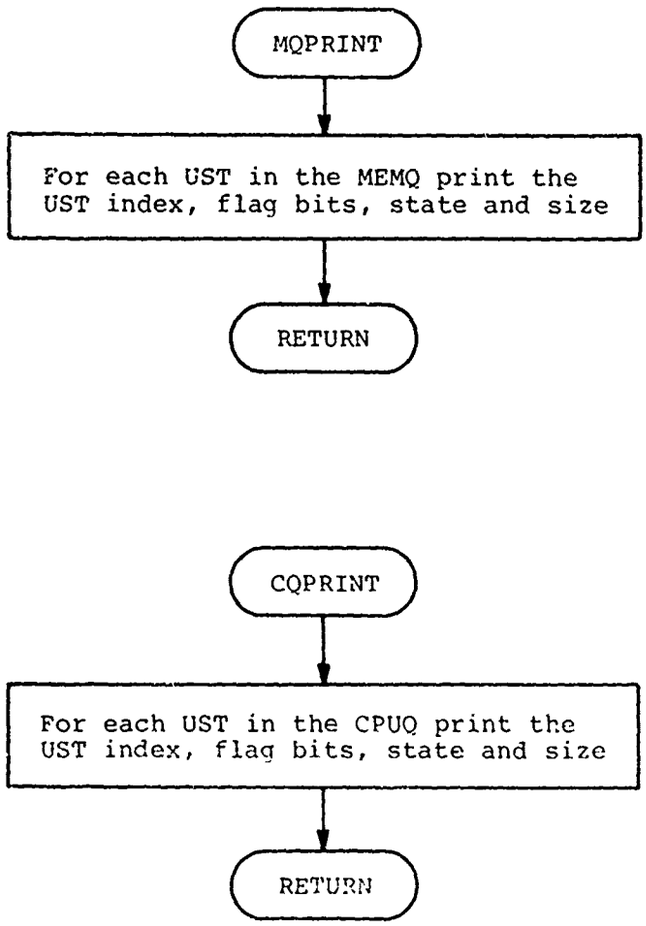
Appendix 2

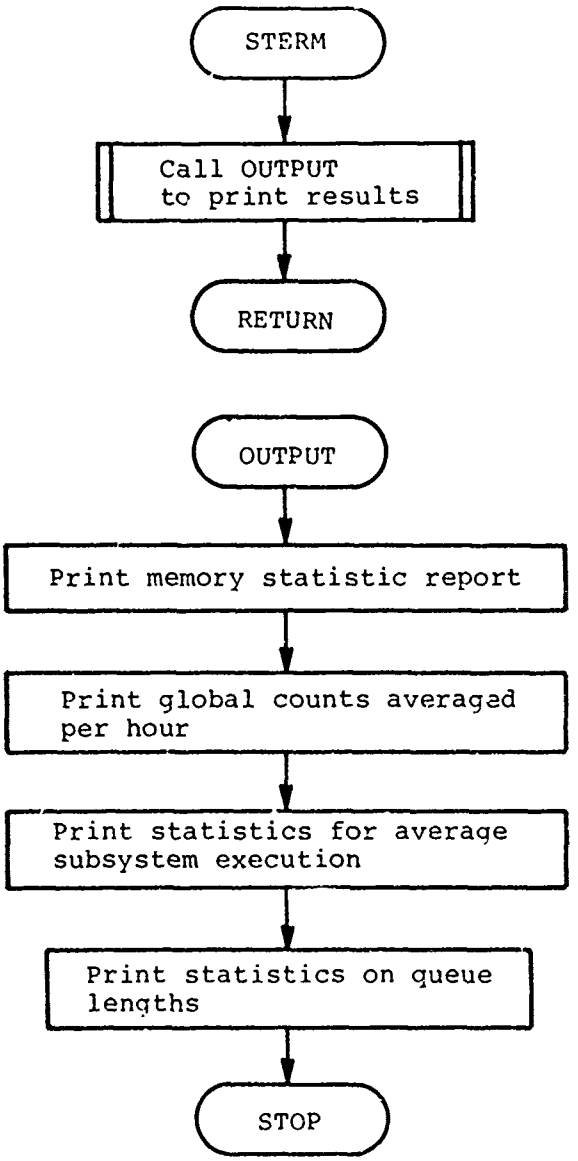
Flowcharts

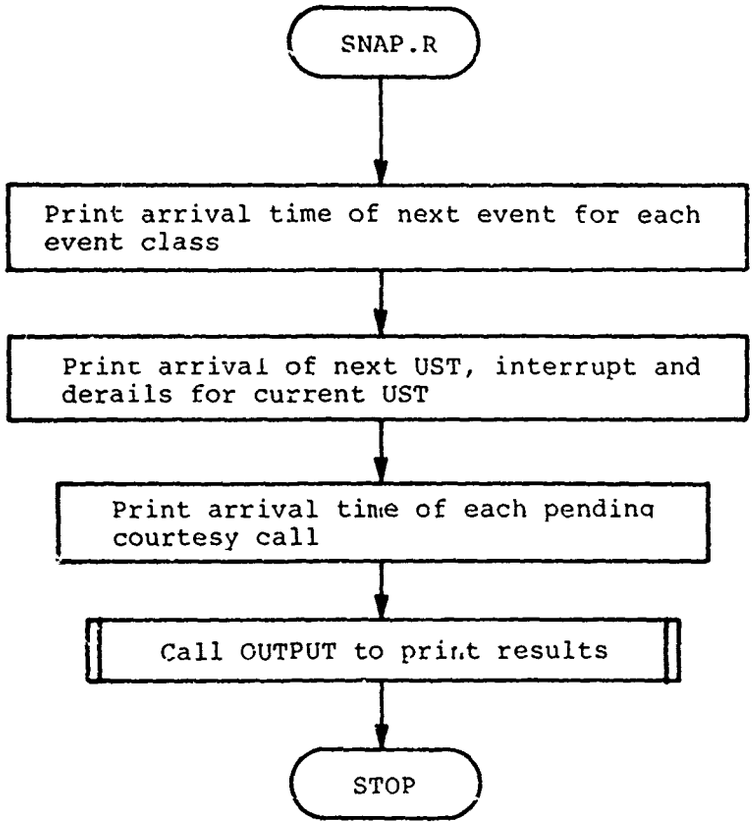

INIT
Read input parameters
Read CHECK
CHECK = 9999 ?
no
Print error message
STOP
yes
Reserve arrays for memory map
Initialize memory map
Sample next arrival time of subsystem, and interrupt
Schedule entry to allocator, simulation termination, diagnostic routine and reinitialization event
RETURN

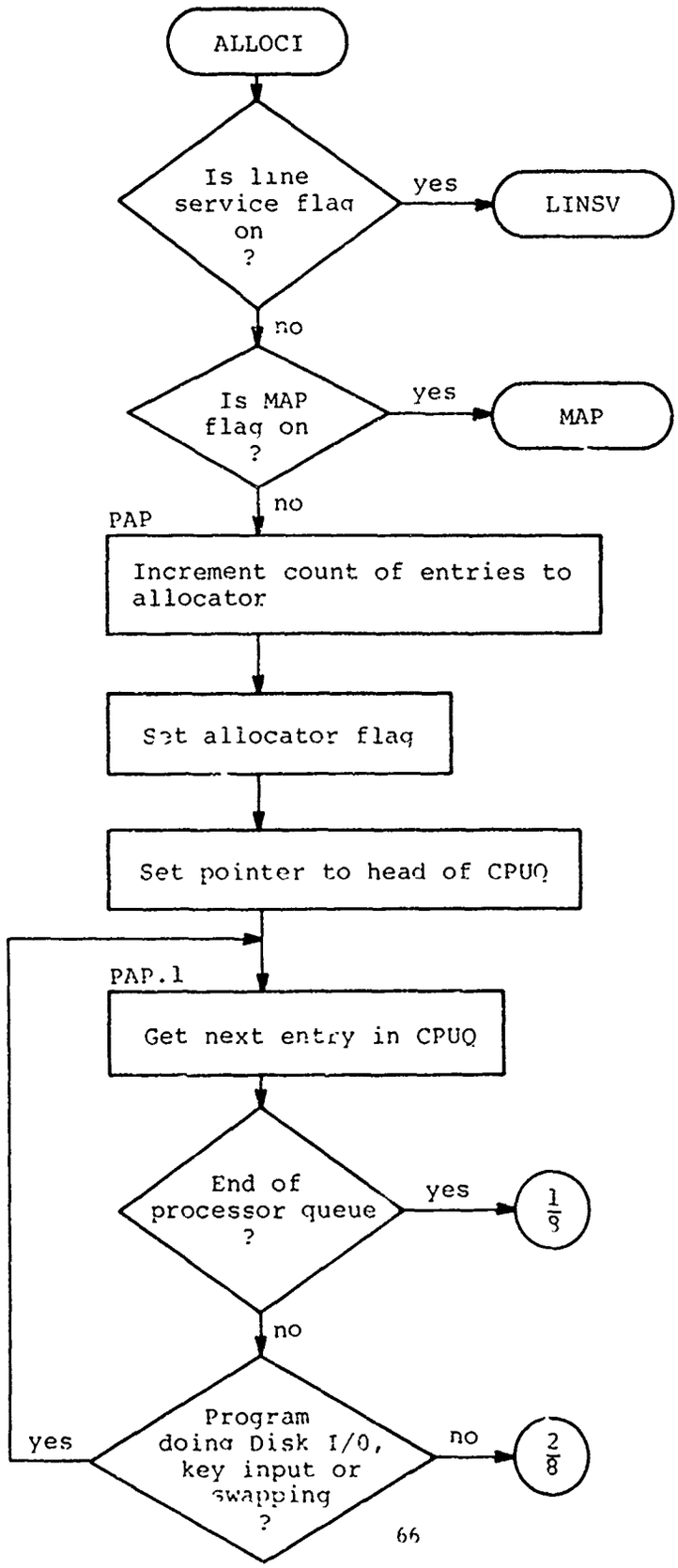

MAIN
Call INIT and then release the routine
Start the simulation clock and proceed to the first event
STOP
PDIAG
Is this the first call to PDIAG ?
yes
no
Set internal switches to the values read in routine INIT
Clear all diagnostic switches
Reschedule this event after the diagnostic duration period
Destroy the PDIAG event notice
RETURN

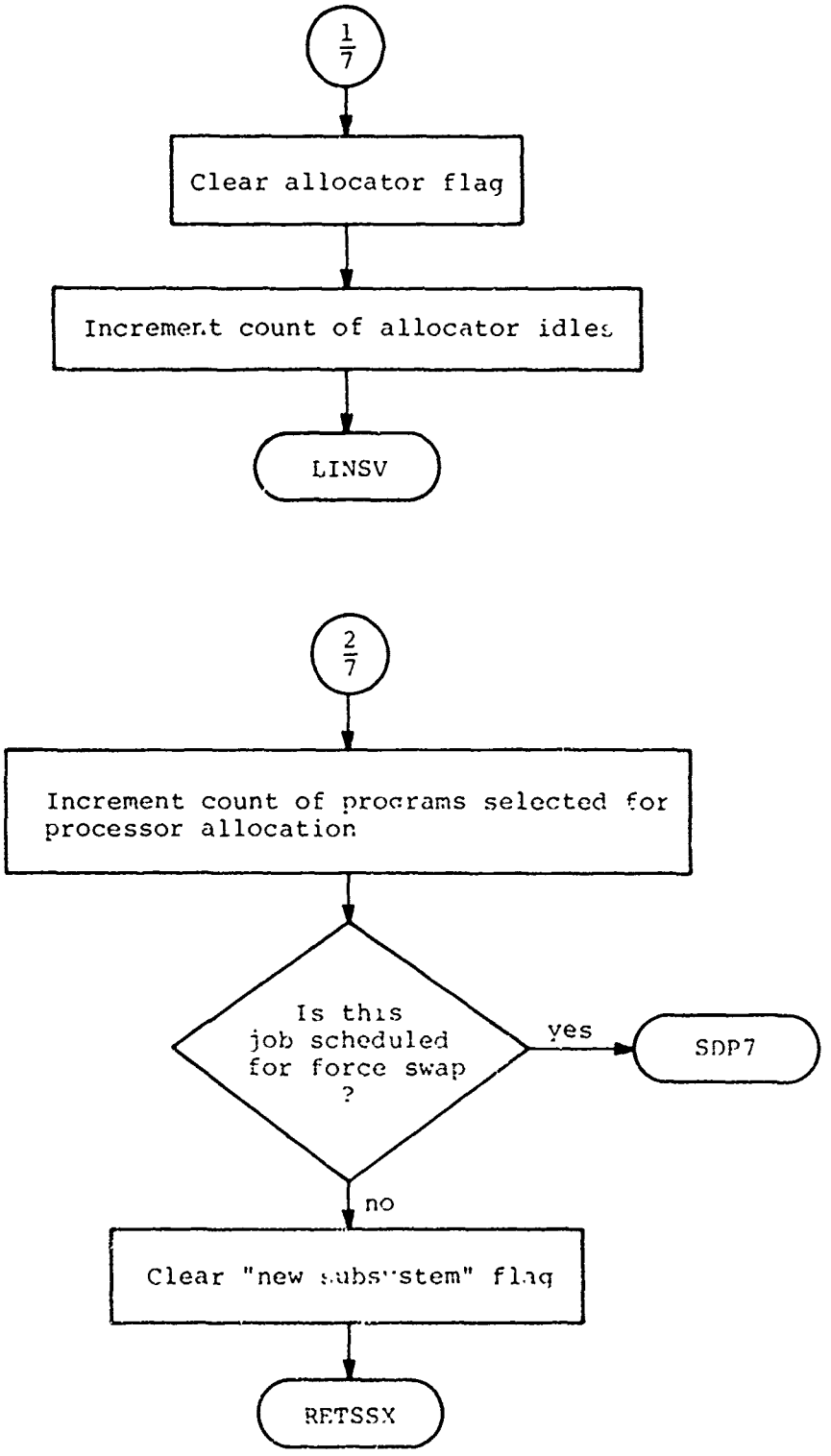

REINIT
Call the Simscript generated routines to reset all counters for ACCUMULATE and TALLY variables
Reset all explicit counters to zero
RETURN
CORSAMP
Sample the program size and hole size of all entries in the memory map. (Except the program size of the first dummy entry.)
Add program sizes in TAUSE and program and hole sizes in TOTCOR
Sample USEDSIZE as TAUSE
Calculate percent core used
RETURN

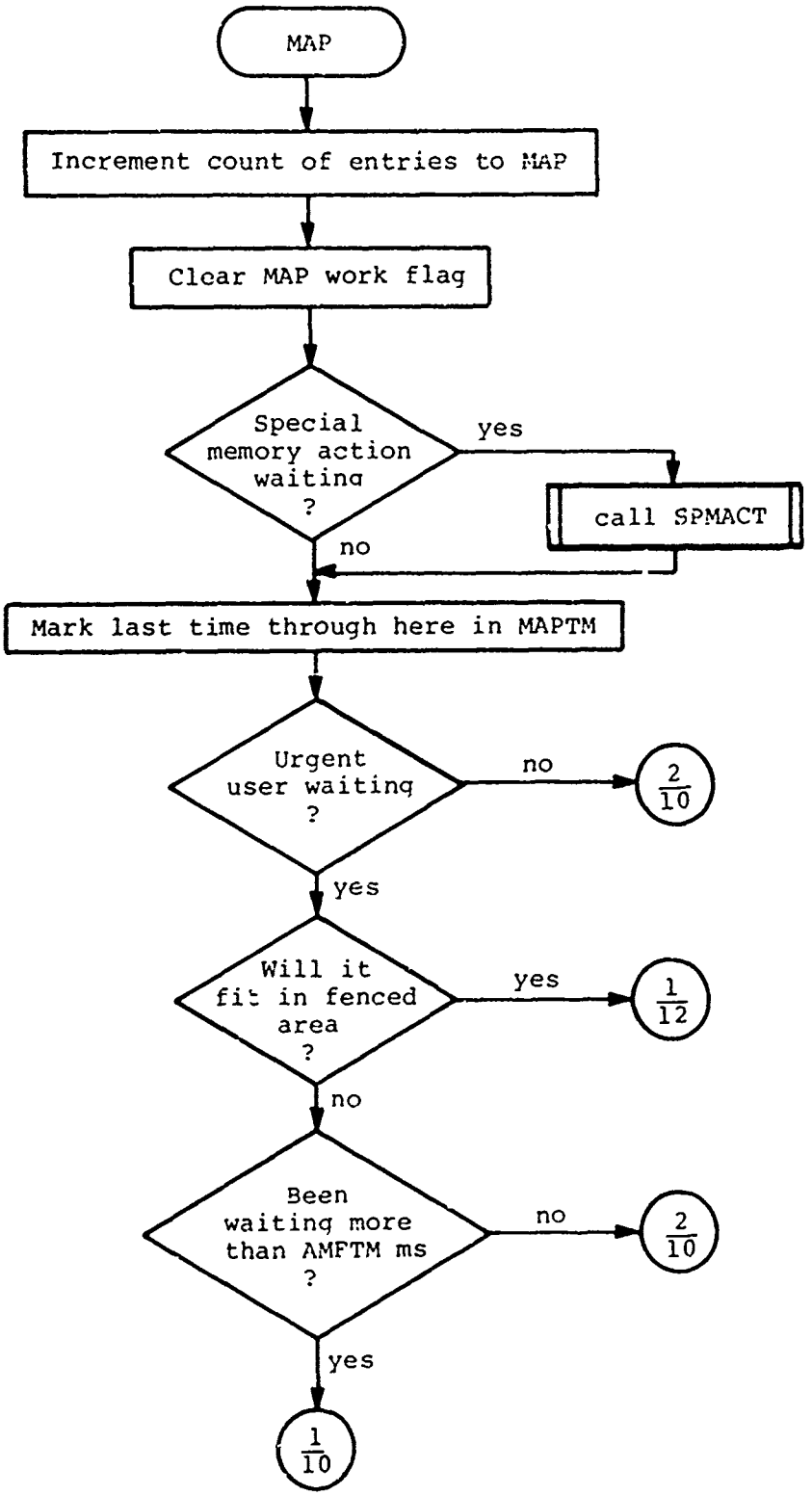

MQPRINT
For each UST in the MEMQ print the UST index, flag bits, state and size
RETURN
CQPRINT
For each UST in the CPUQ print the UST index, flag bits, state and size
RETURN

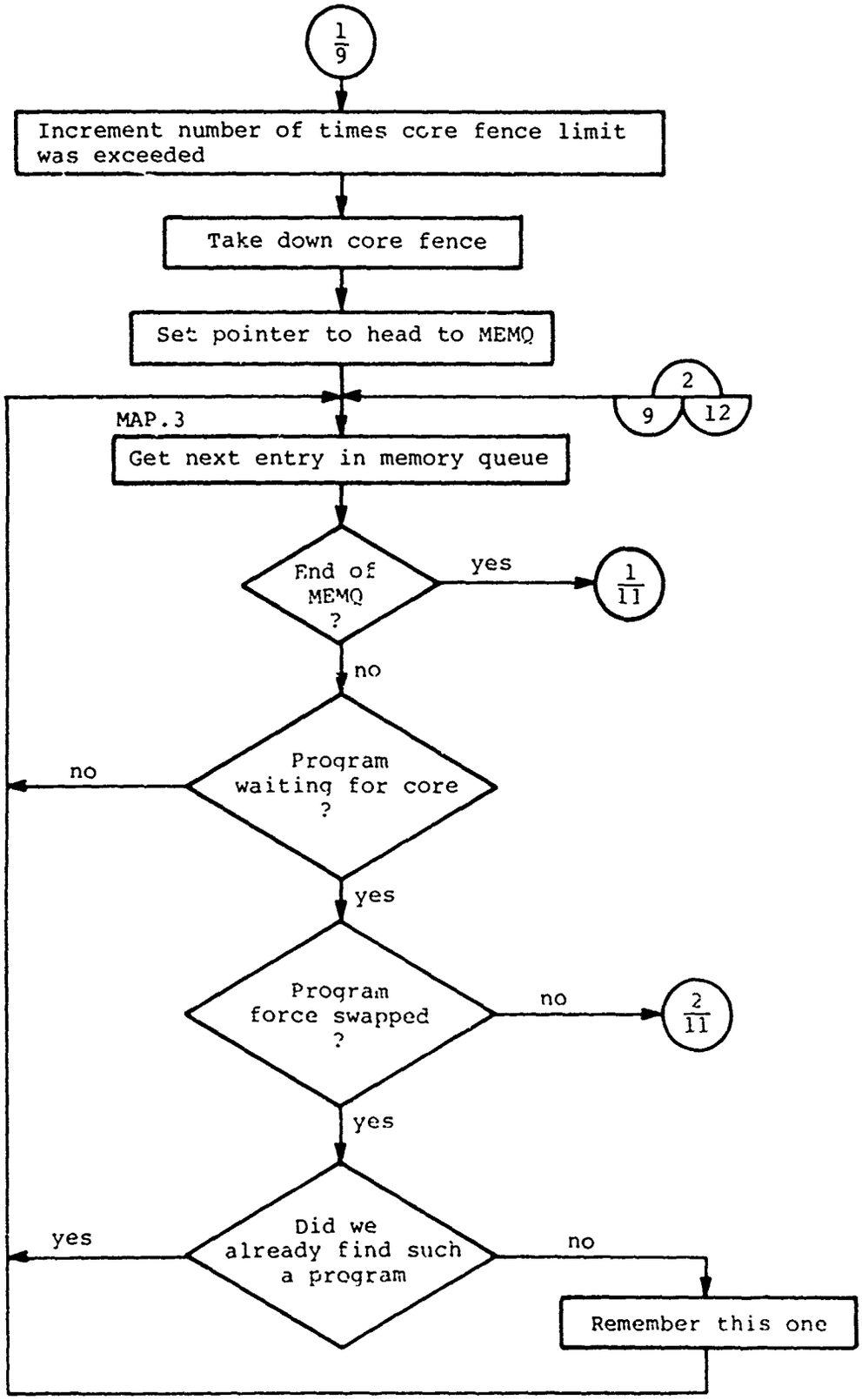

STERM
Call OUTPUT to print results
RETURN
OUTPUT
Print memory statistic report
Print global counts averaged per hour
Print statistics for average subsystem execution
Print statistics on queue lengths
STOP

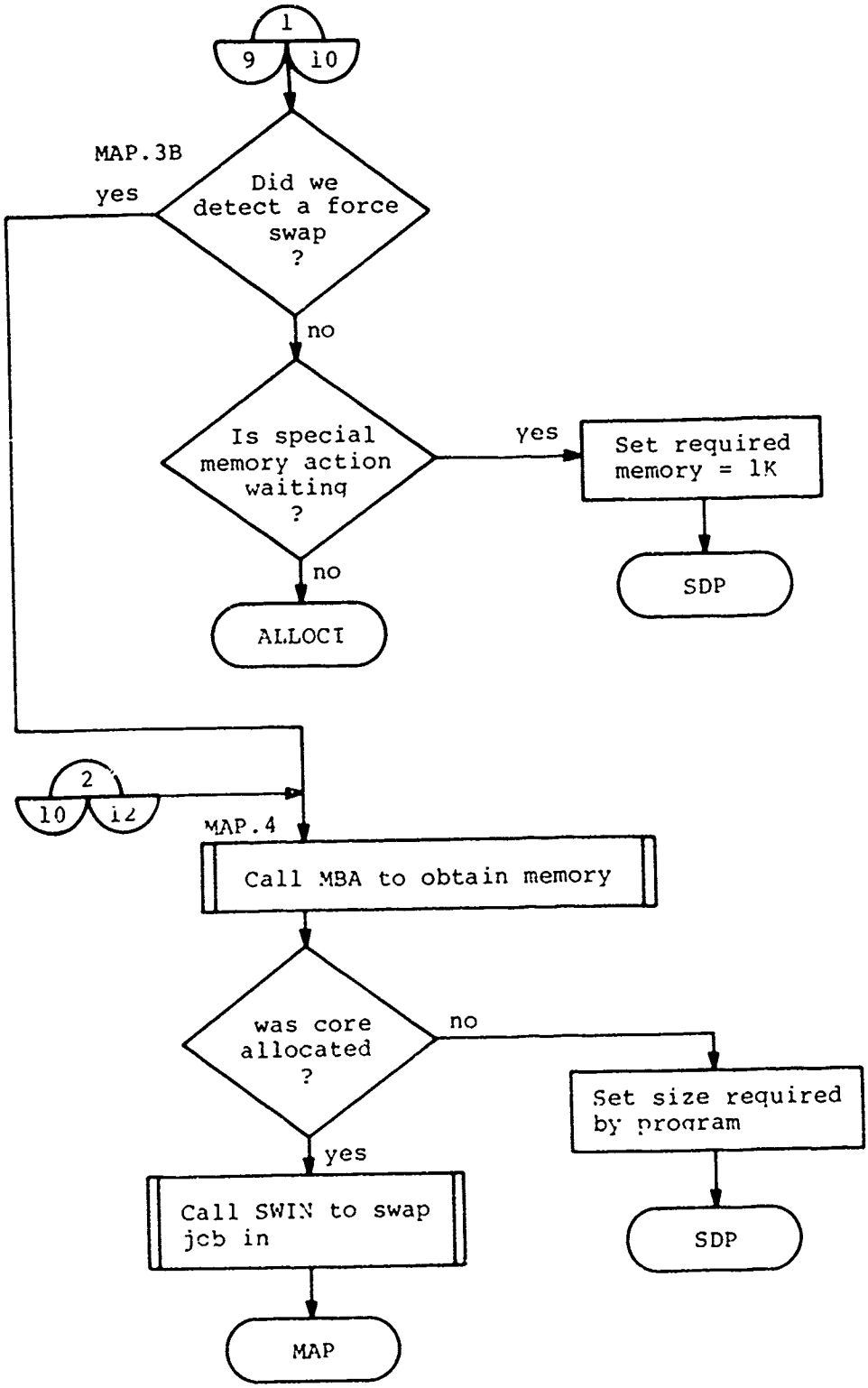

SNAP.R
Print arrival time of next event for each event class
Print arrival of next UST, interrupt and derails for current UST
Print arrival time of each pending courtesy call
Call OUTPUT to print results
STOP

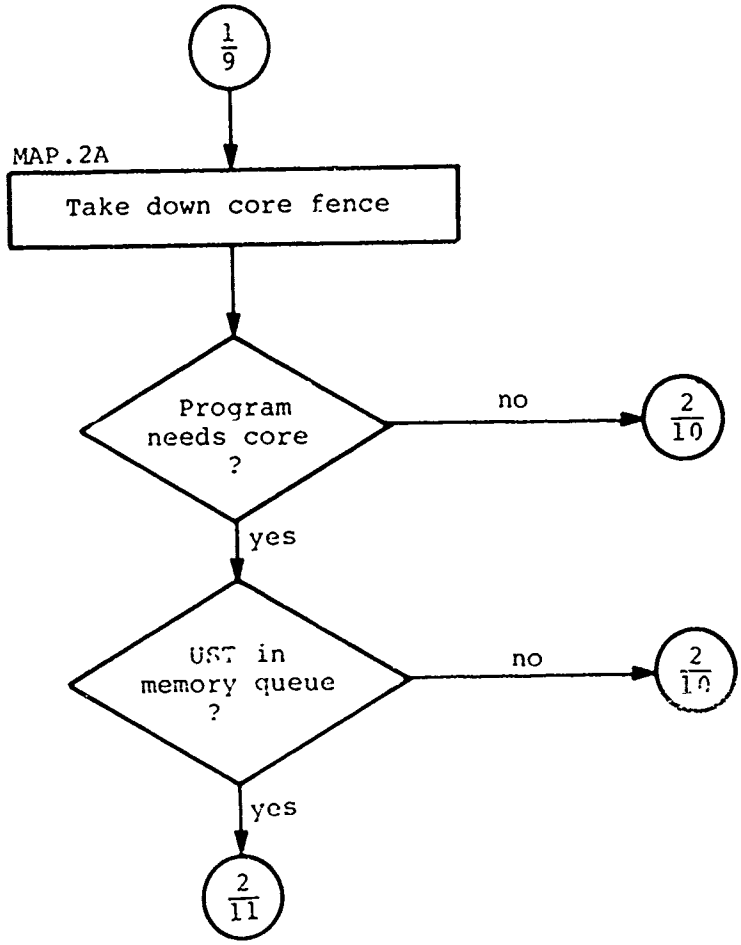

ALLOCI
Is line service flag on ?
yes
LINSV
no
Is MAP flag on ?
yes
MAP
no
PAP
Increment count of entries to allocator
Set allocator flag
Set pointer to head of CPUQ
PAP.1
Get next entry in CPUQ
End of processor queue ?
yes
1/9
no
Program doing Disk I/O, key input or swapping ?
yes
no
2/8

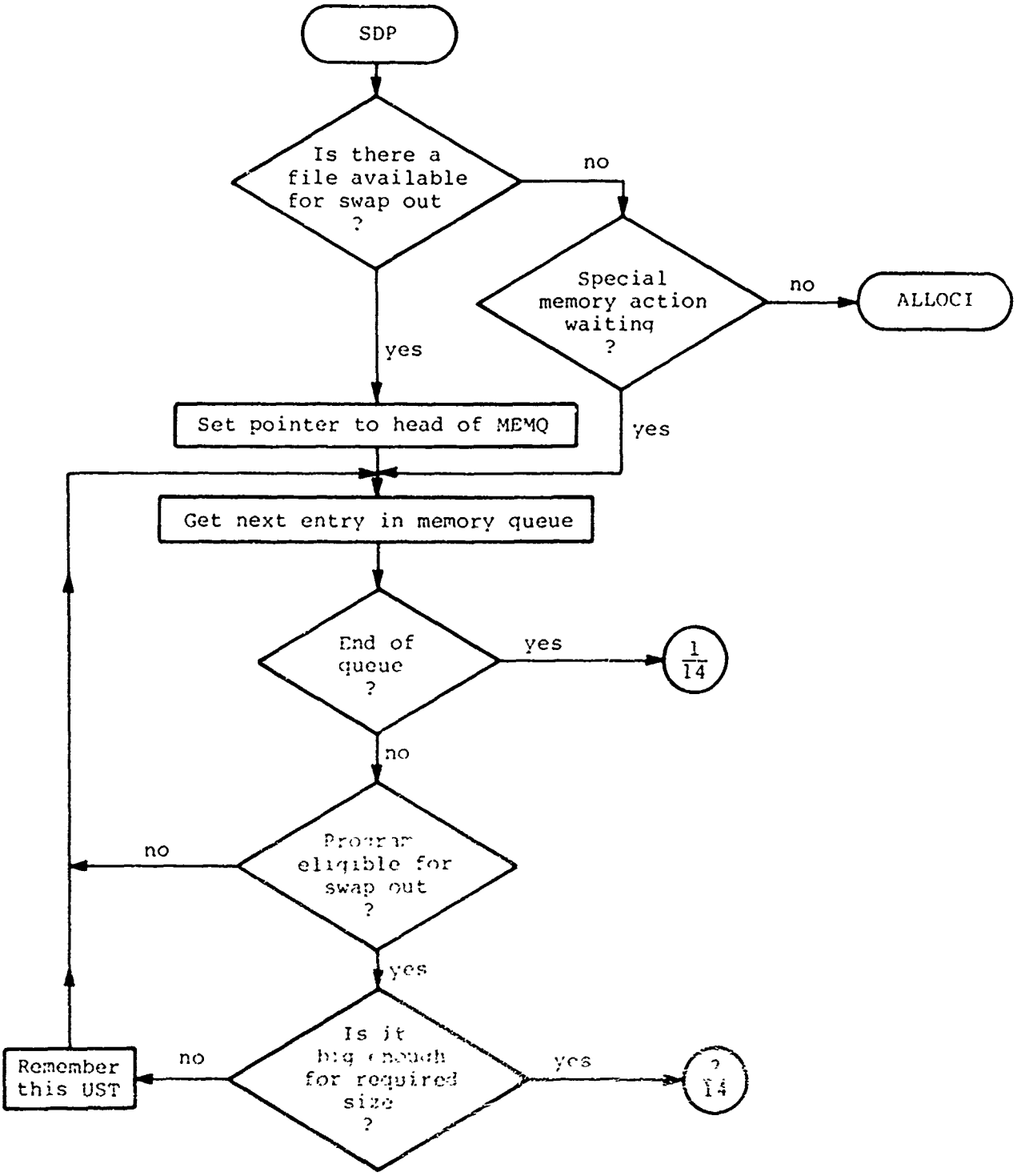

1/7
Clear allocator flag
Increment count of allocator idles
LINSV
2/7
Increment count of programs selected for processor allocation
Is this job scheduled for force swap ?
yes
SDP7
no
Clear "new subsystem" flag
RETSSX

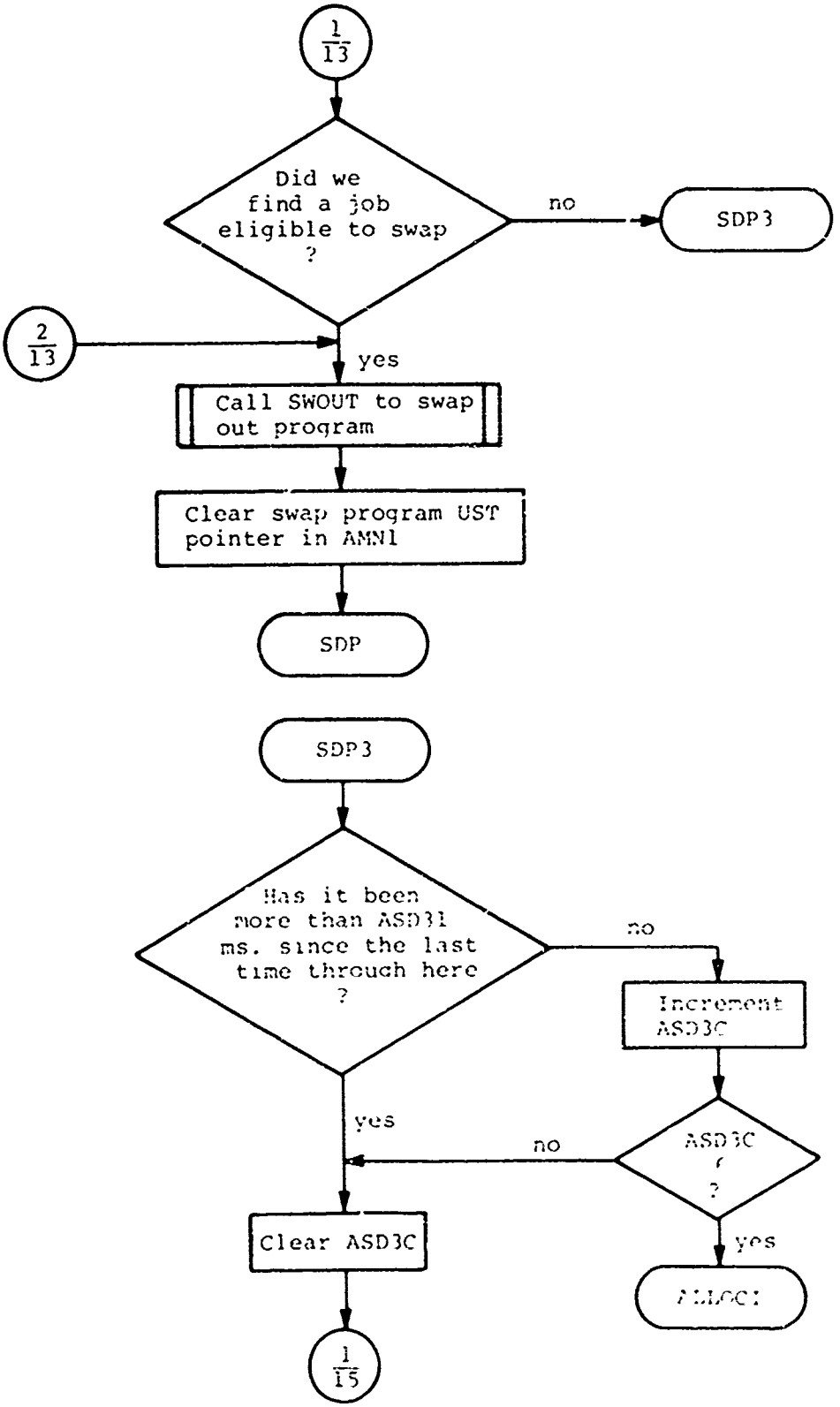

MAP
Increment count of entries to MAP
Clear MAP work flag
Special memory action waiting ?
yes
call SPMACT
no
Mark last time through here in MAPTM
Urgent user waiting ?
no
2/10
yes
Will it fit in fenced area ?
yes
1/12
no
Been waiting more than AMFTM ms ?
no
2/10
yes
1/10

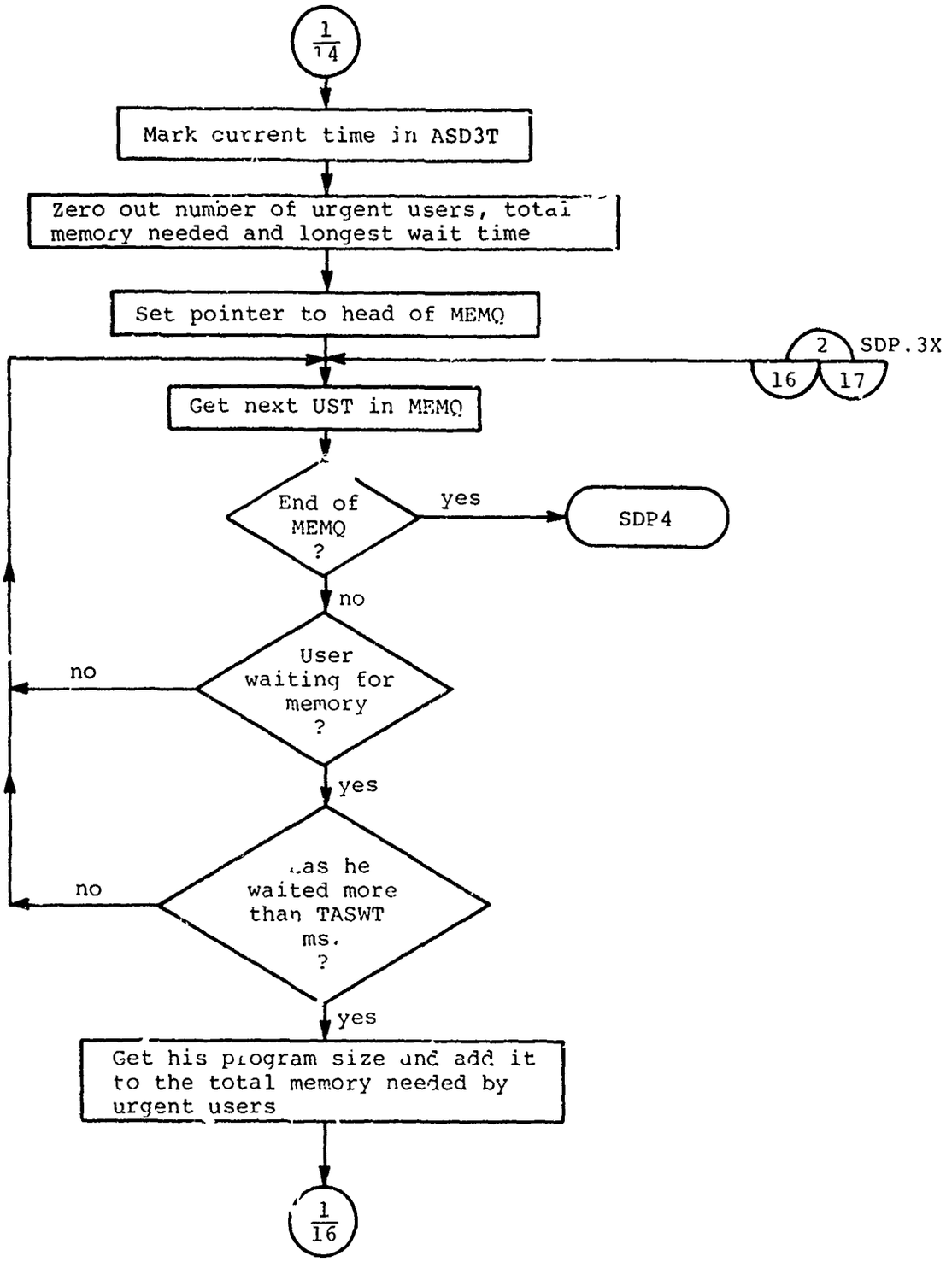

1/9
Increment number of times core fence limit was exceeded
Take down core fence
Set pointer to head to MEMQ
2/9
2/12
MAP.3
Get next entry in memory queue
End of MEMQ ?
yes
1/11
no
Program waiting for core ?
no
yes
Program force swapped ?
no
2/11
yes
Did we already find such a program
yes
no
Remember this one

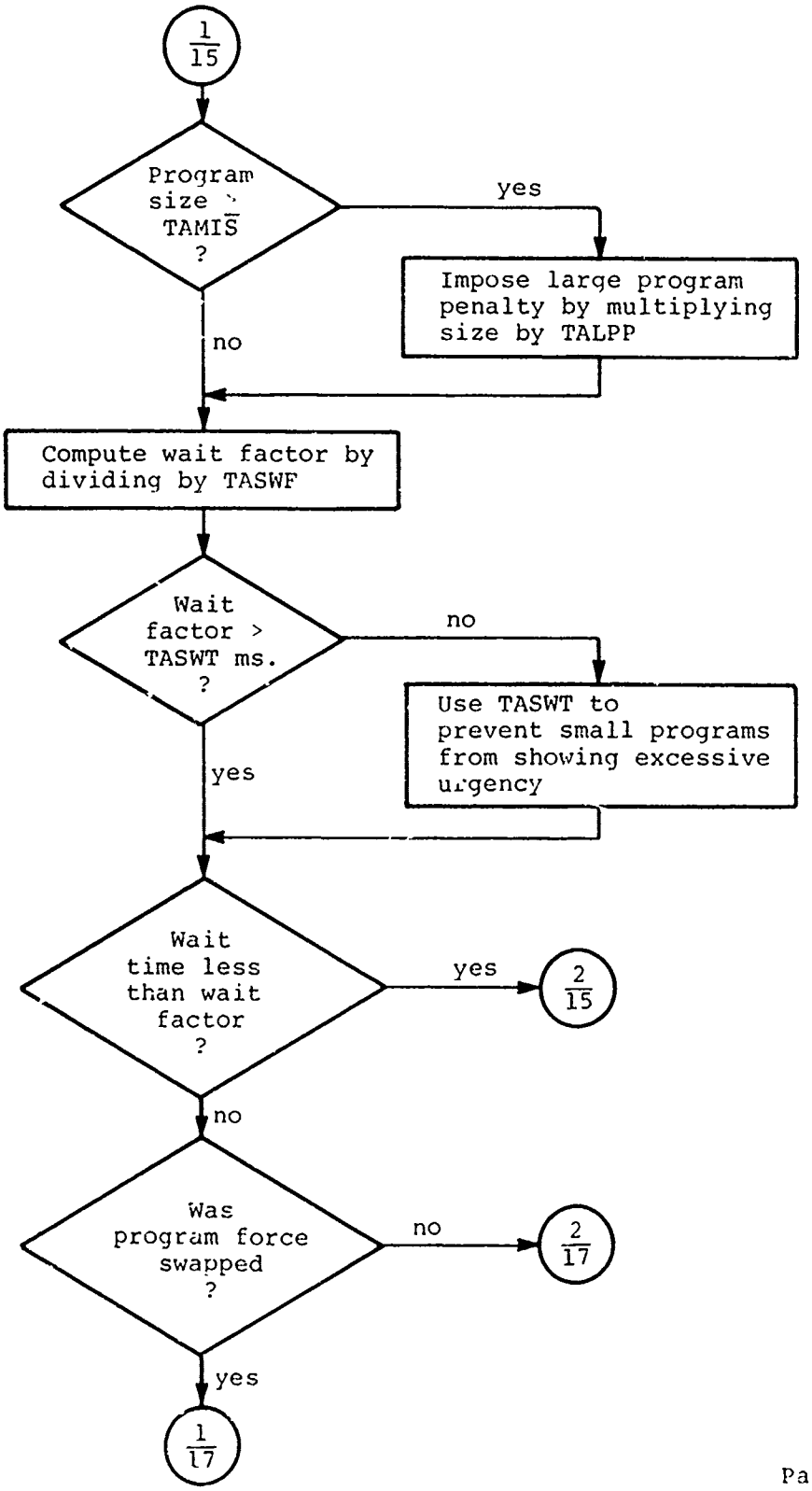

1
9
10
MAP.3B
yes
Did we detect a force swap ?
no
Is special memory action waiting ?
yes
Set required memory = 1K
SDP
no
ALLOCI
2
10
12
MAP.4
Call MBA to obtain memory
was core allocated ?
no
Set size required by program
SDP
yes
Call SWIN to swap jcb in
MAP

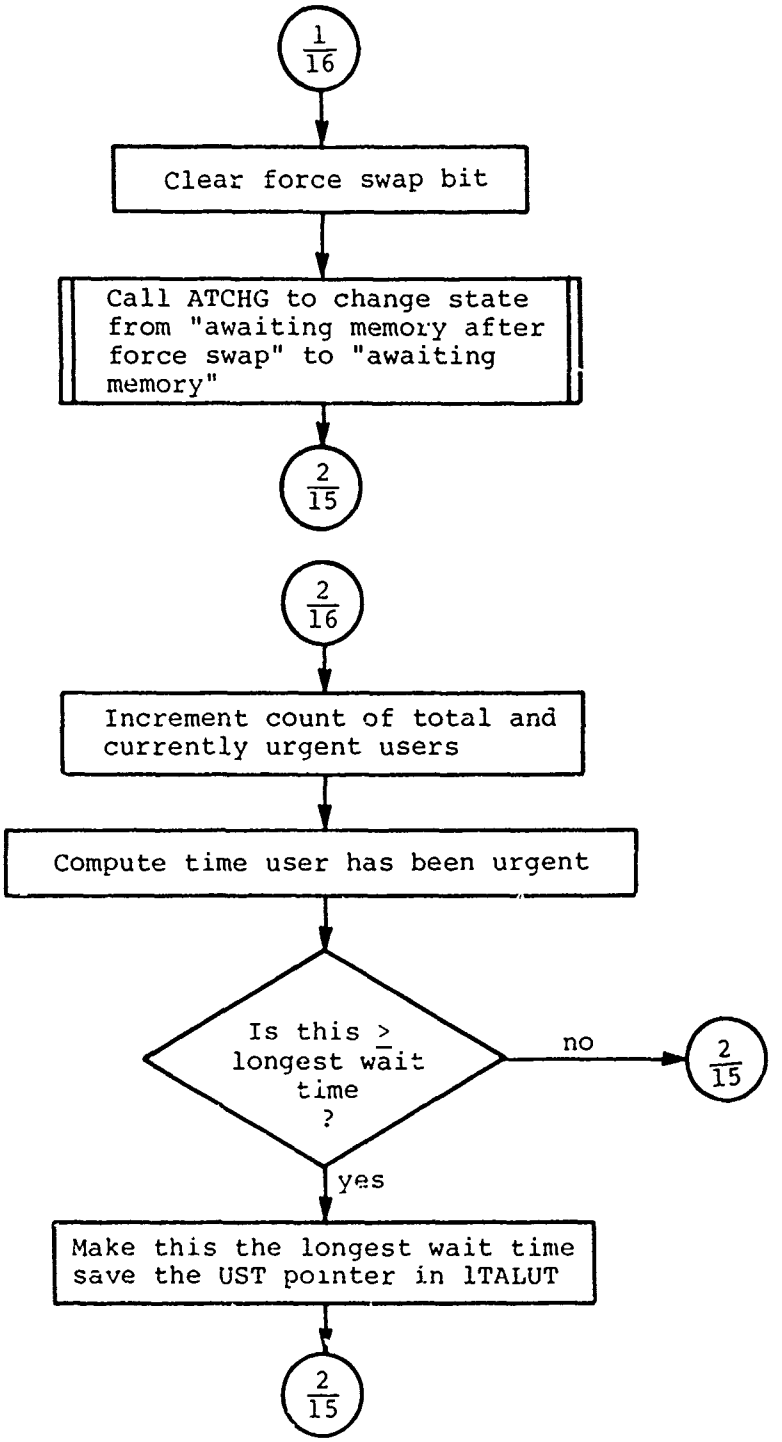

1/9
MAP.2A
Take down core fence
Program needs core ?
no
2/10
yes
UST in memory queue ?
no
2/10
yes
2/11

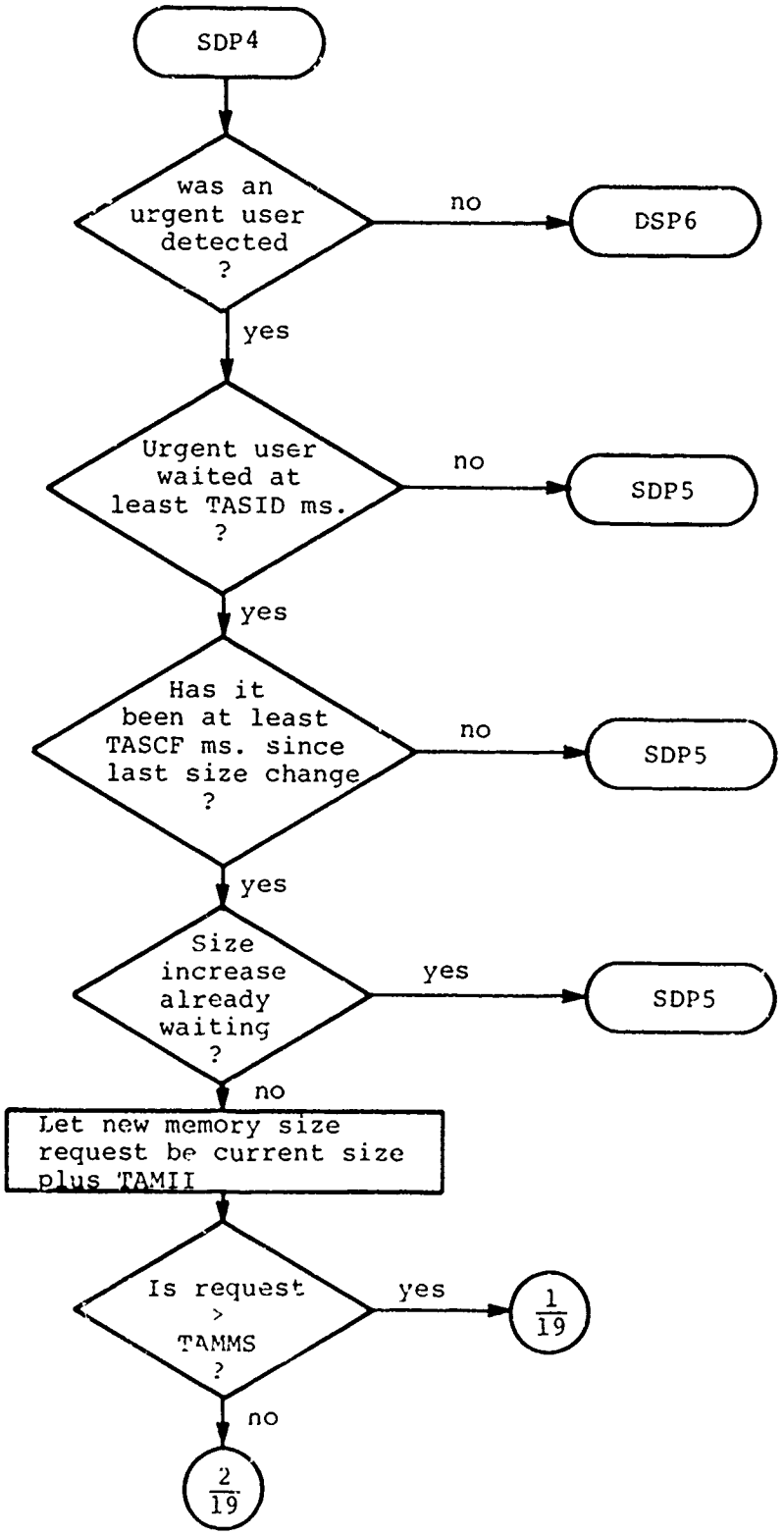

SDP
Is there a file available for swap out ?
no
Special memory action waiting ?
no
ALLOCI
yes
yes
Set pointer to head of MEMQ
Get next entry in memory queue
End of queue ?
yes
1/14
no
Program eligible for swap out ?
no
yes
Is it big enough for required size ?
no
yes
Remember this UST
?/14

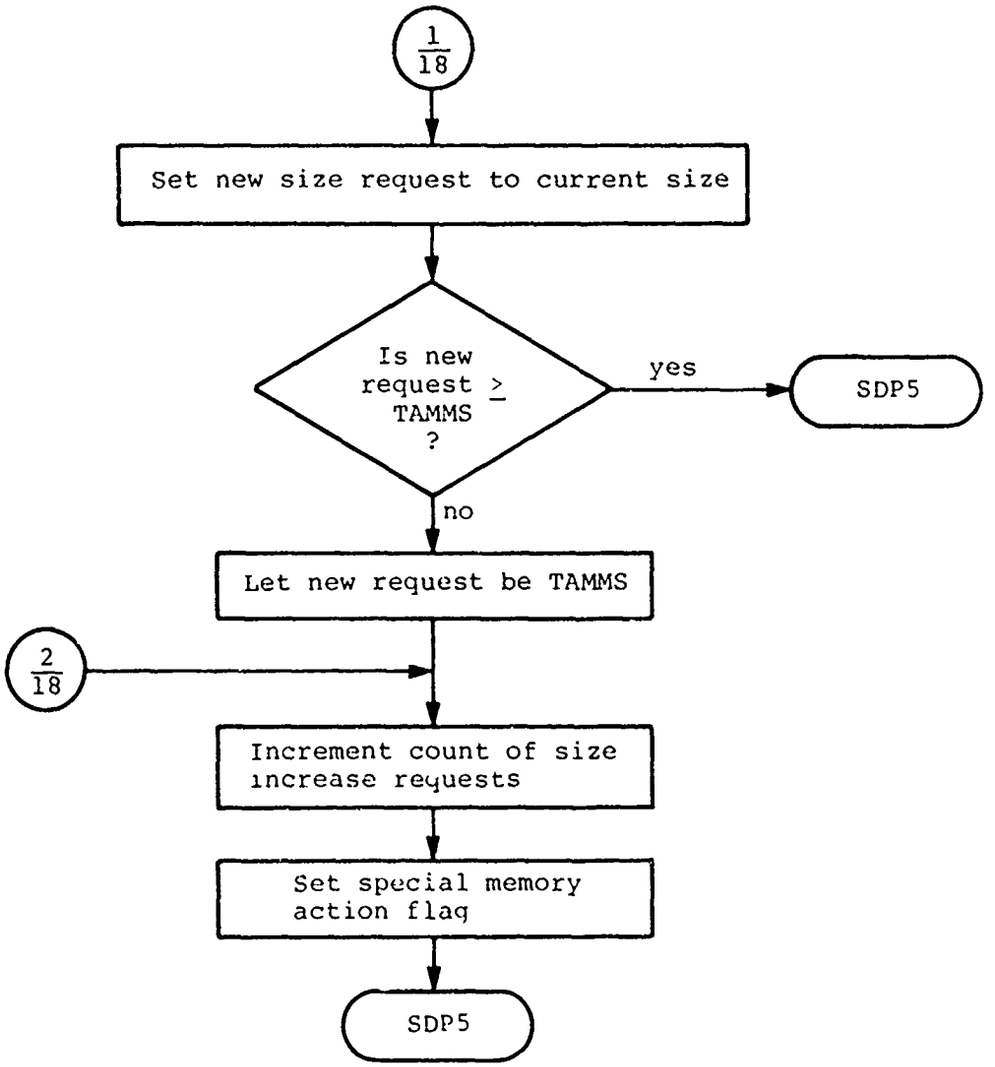

1/13
Did we find a job eligible to swap ?
no
SDP3
2/13
yes
Call SWOUT to swap out program
Clear swap program UST pointer in AMN1
SDP
SDP3
Has it been more than ASD31 ms. since the last time through here ?
no
Increment ASD3C
yes
no
ASD3C ?
yes
Clear ASD3C
1/15

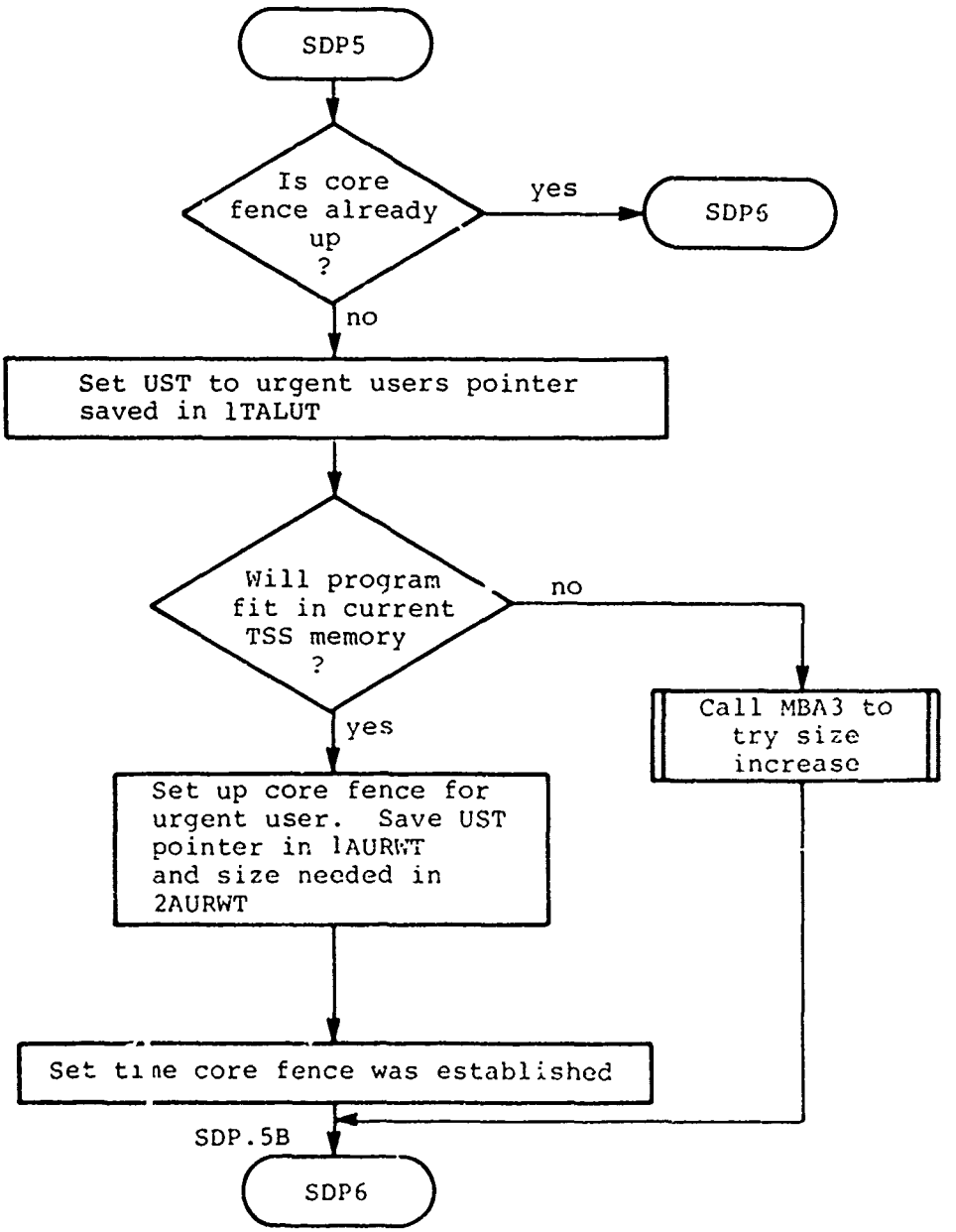

1/14
Mark current time in ASD3T
Zero out number of urgent users, total memory needed and longest wait time
Set pointer to head of MEMQ
2
16
17
SDP.3X
Get next UST in MEMQ
End of MEMQ ?
yes
SDP4
no
User waiting for memory ?
no
yes
..as he waited more than TASWT ms. ?
no
yes
Get his program size and add it to the total memory needed by urgent users
1/16

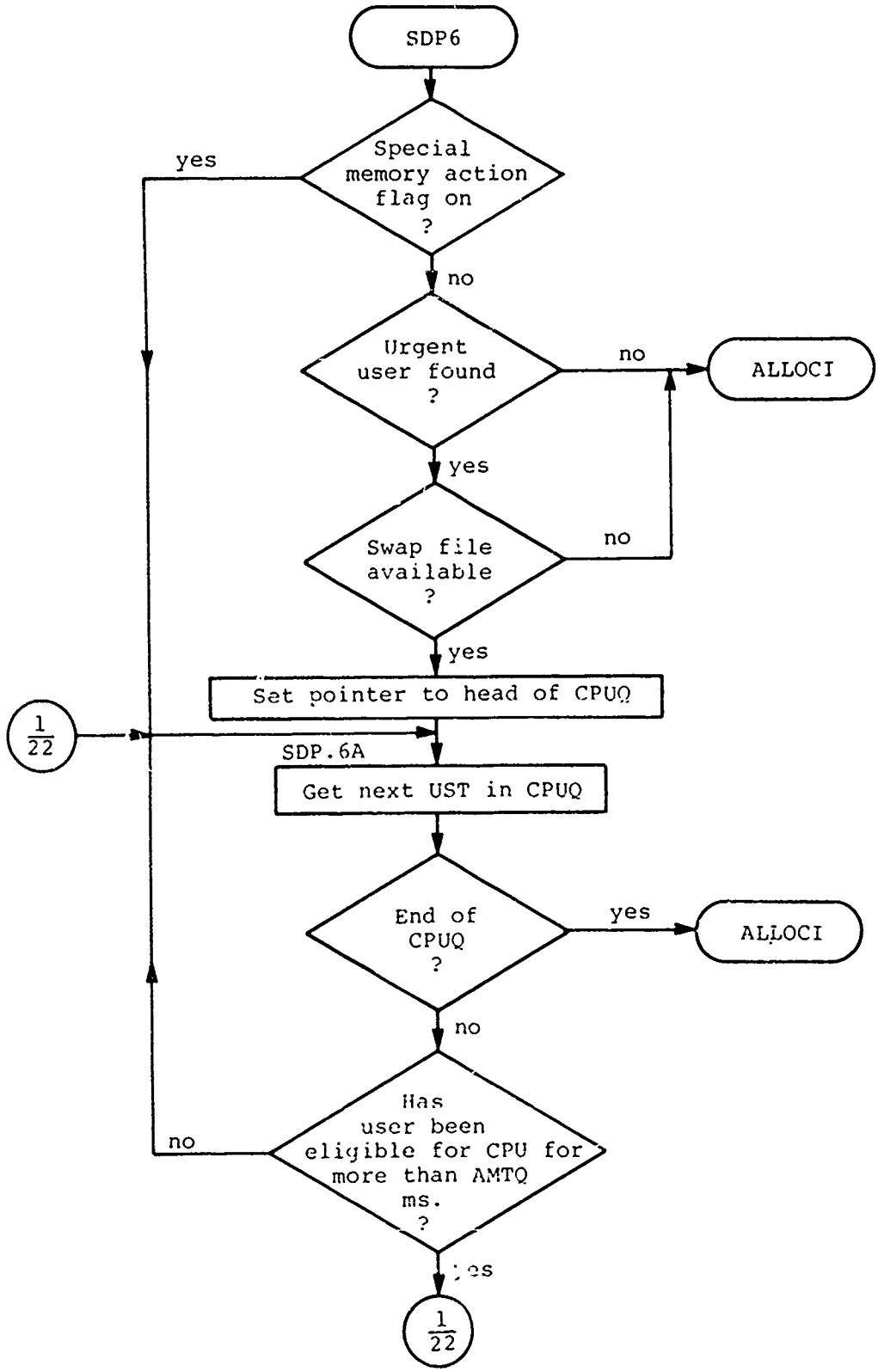

1/15
Program size > TAMIS ?
yes
Impose large program penalty by multiplying size by TALPP
no
Compute wait factor by dividing by TASWF
Wait factor > TASWT ms. ?
no
Use TASWT to prevent small programs from showing excessive urgency
yes
Wait time less than wait factor ?
yes
2/15
no
Was program force swapped ?
no
2/17
yes
1/17

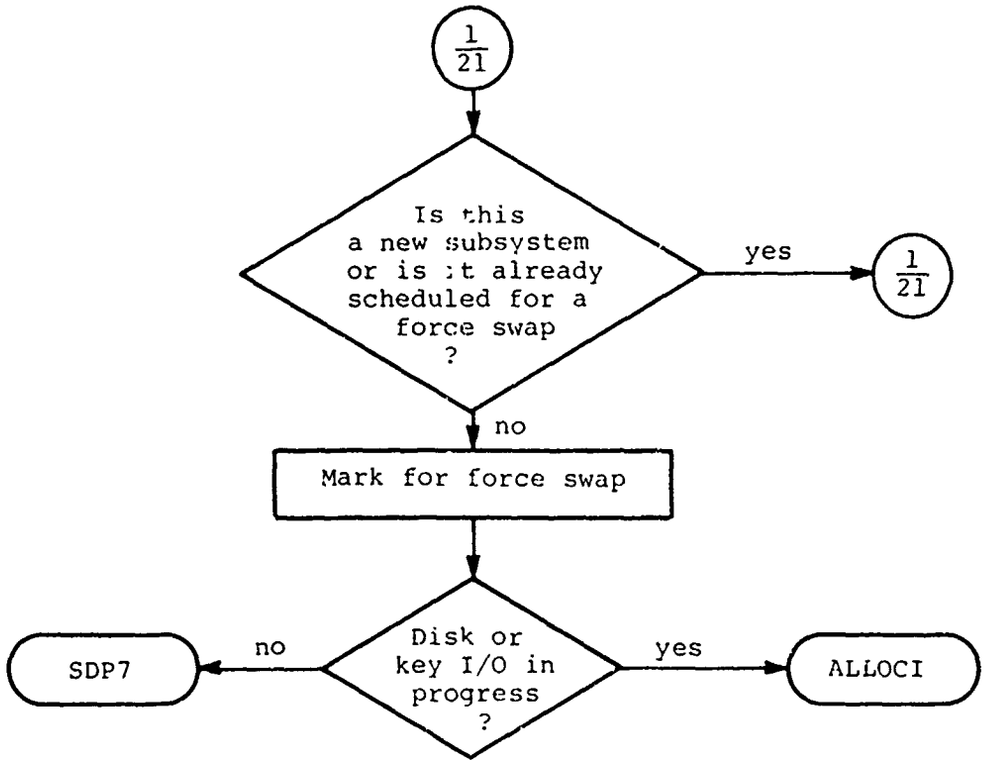

1/16
Clear force swap bit
Call ATCHG to change state from "awaiting memory after force swap" to "awaiting memory"
2/15
2/16
Increment count of total and currently urgent users
Compute time user has been urgent
Is this ≥ longest wait time ?
no
2/15
yes
Make this the longest wait time save the UST pointer in lTALUT
2/15

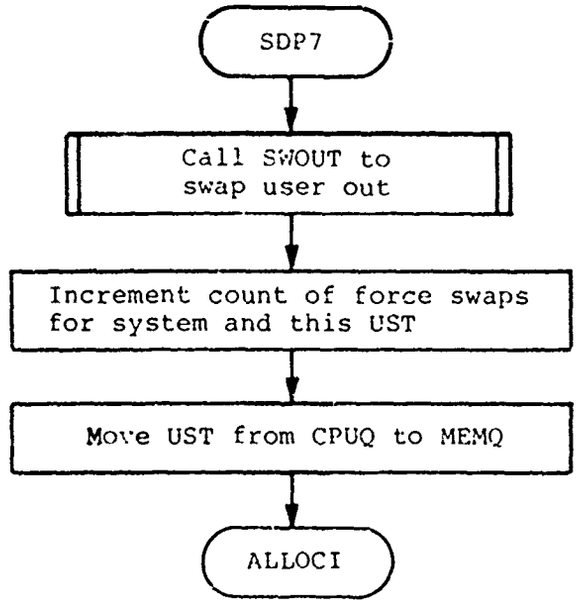

SDP4
was an urgent user detected ?
no
DSP6
yes
Urgent user waited at least TASID ms. ?
no
SDP5
yes
Has it been at least TASCF ms. since last size change ?
no
SDP5
yes
Size increase already waiting ?
yes
SDP5
no
Let new memory size request be current size plus TAMII
Is request > TAMMS ?
yes
1/19
no
2/19

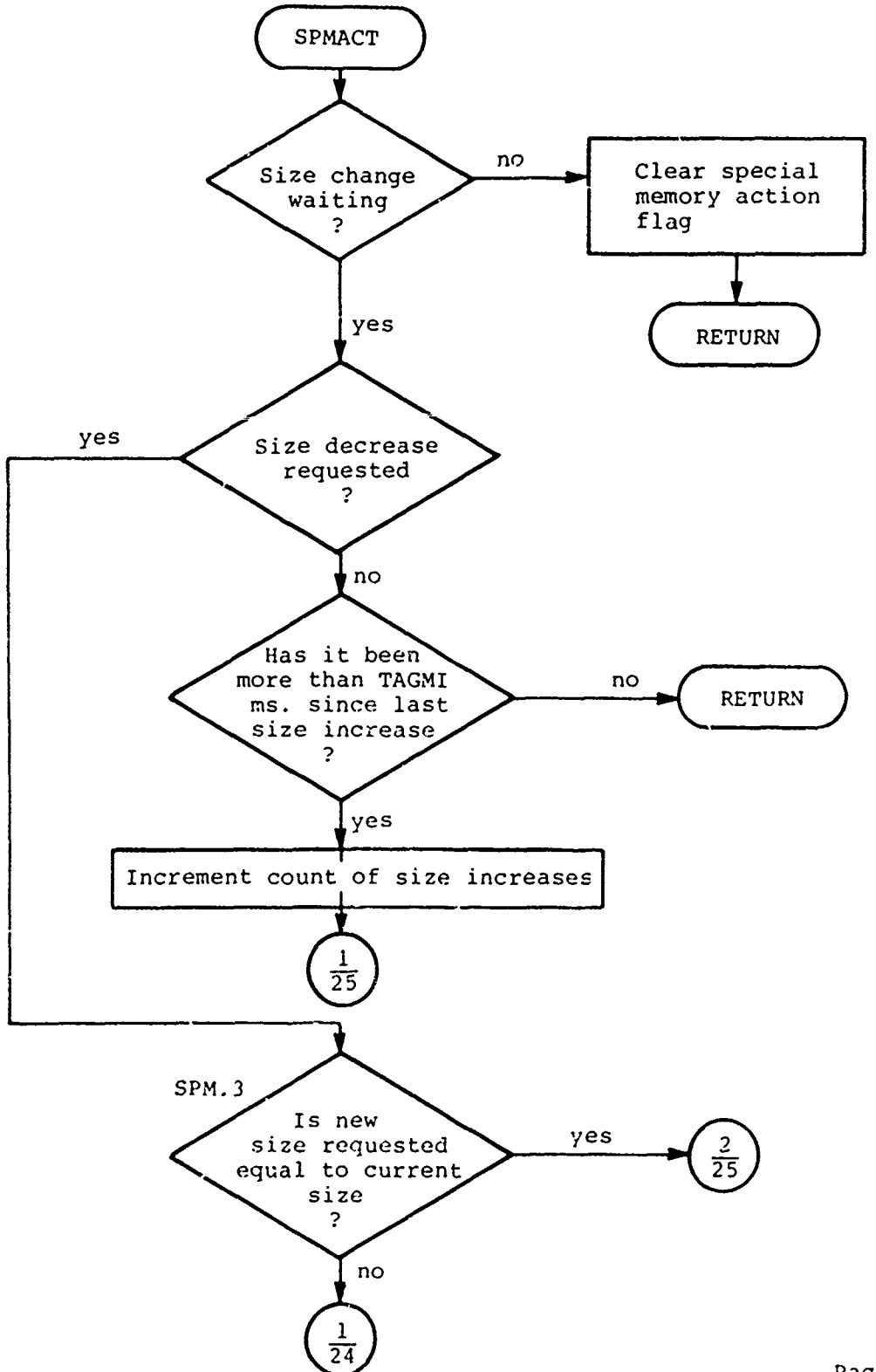

1/18
Set new size request to current size
Is new request ≥ TAMMS ?
yes
SDP5
no
Let new request be TAMMS
2/18
Increment count of size increase requests
Set special memory action flag
SDP5

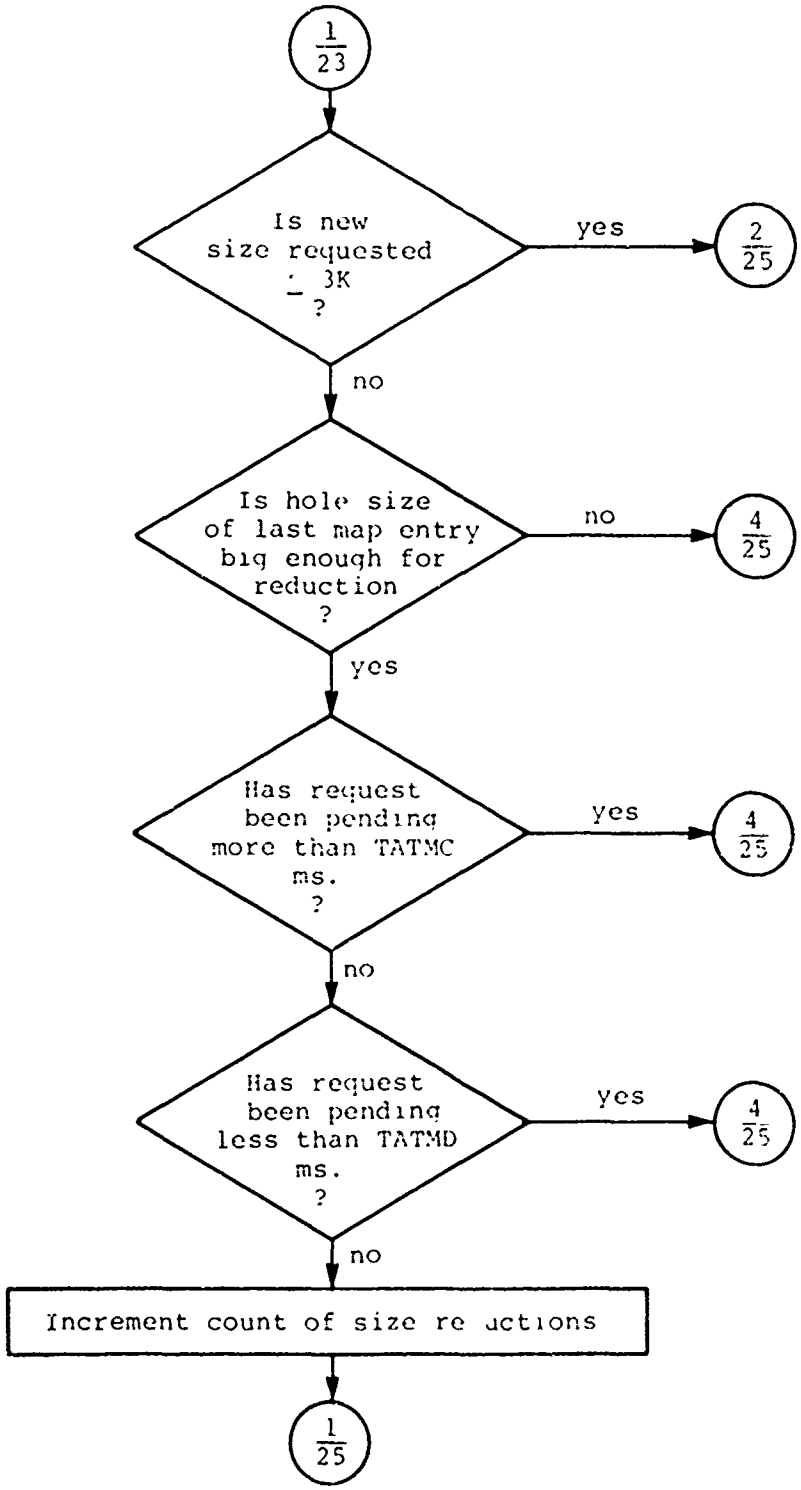

SDP5
Is core fence already up ?
yes
SDP6
no
Set UST to urgent users pointer saved in 1TALUT
Will program fit in current TSS memory ?
no
yes
Call MBA3 to try size increase
Set up core fence for urgent user. Save UST pointer in 1AURWT and size needed in 2AURWT
Set time core fence was established
SDP.5B
SDP6

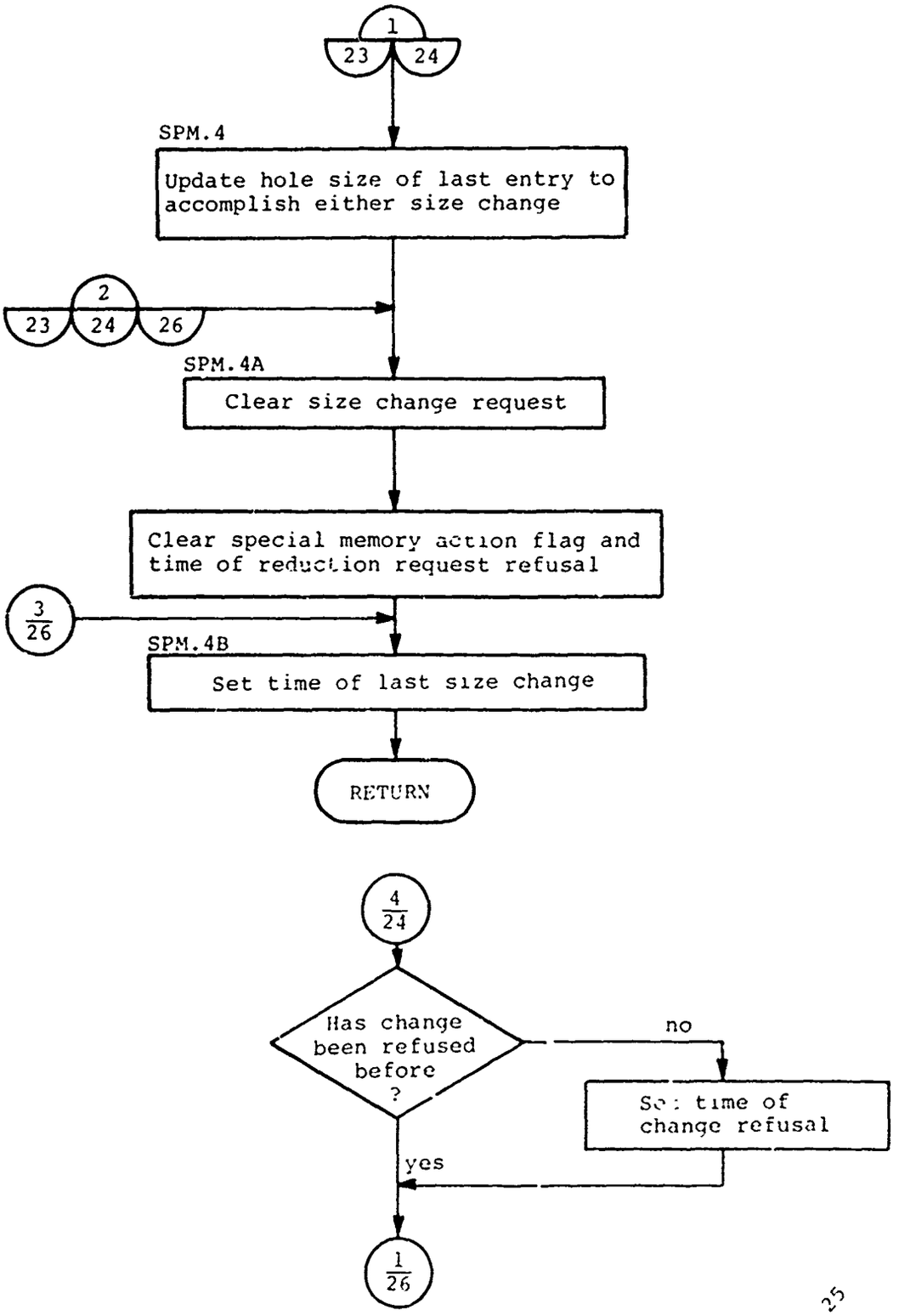

SDP6
Special memory action flag on ?
yes
no
Urgent user found ?
no
ALLOCI
yes
Swap file available ?
no
yes
Set pointer to head of CPUQ
1/22
SDP.6A
Get next UST in CPUQ
End of CPUQ ?
yes
ALLOCI
no
Has user been eligible for CPU for more than AMTQ ms. ?
no
yes
1/22

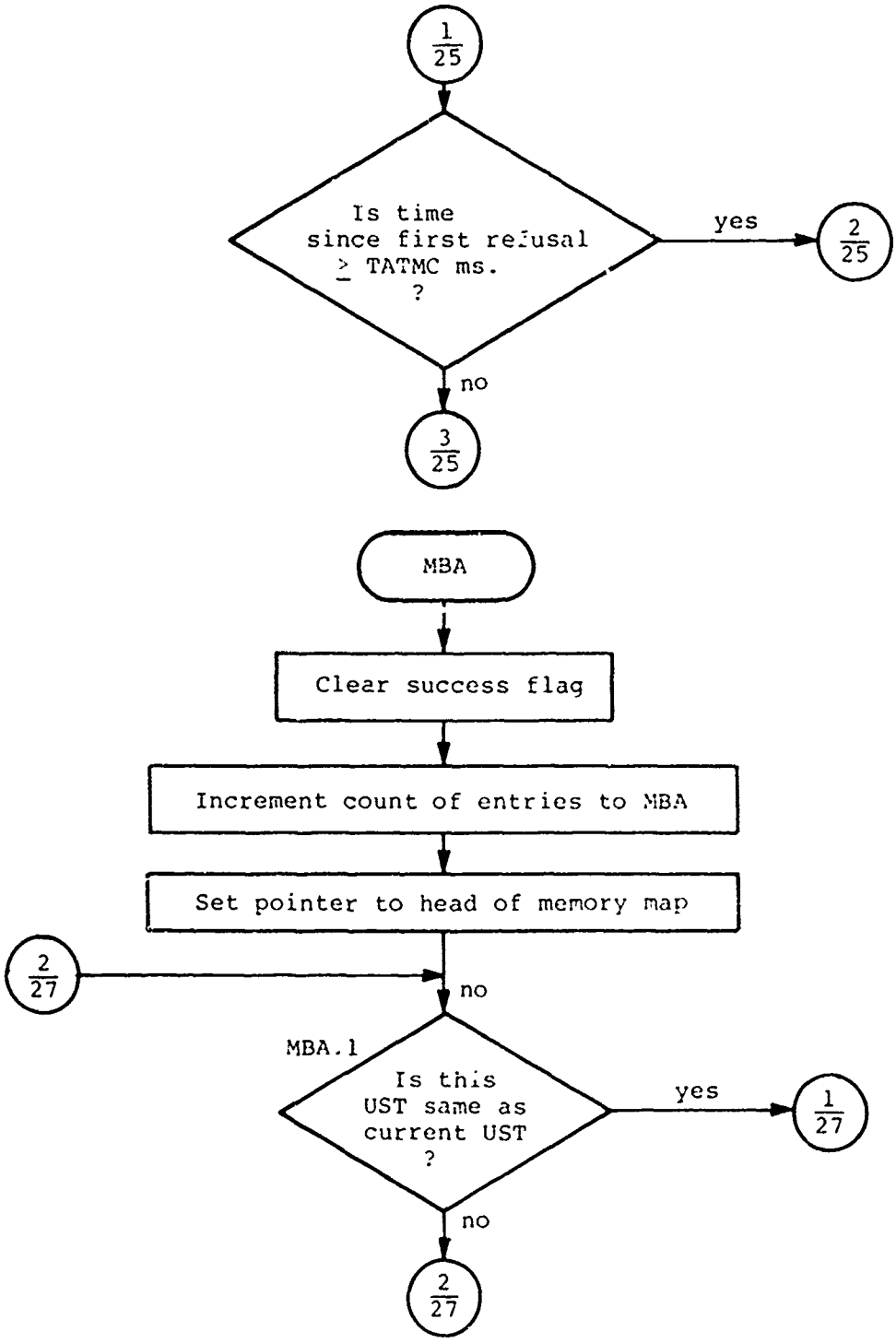

1/21
Is this a new subsystem or is it already scheduled for a force swap ?
yes
1/21
no
Mark for force swap
Disk or key I/O in progress ?
no
SDP7
yes
ALLOCI

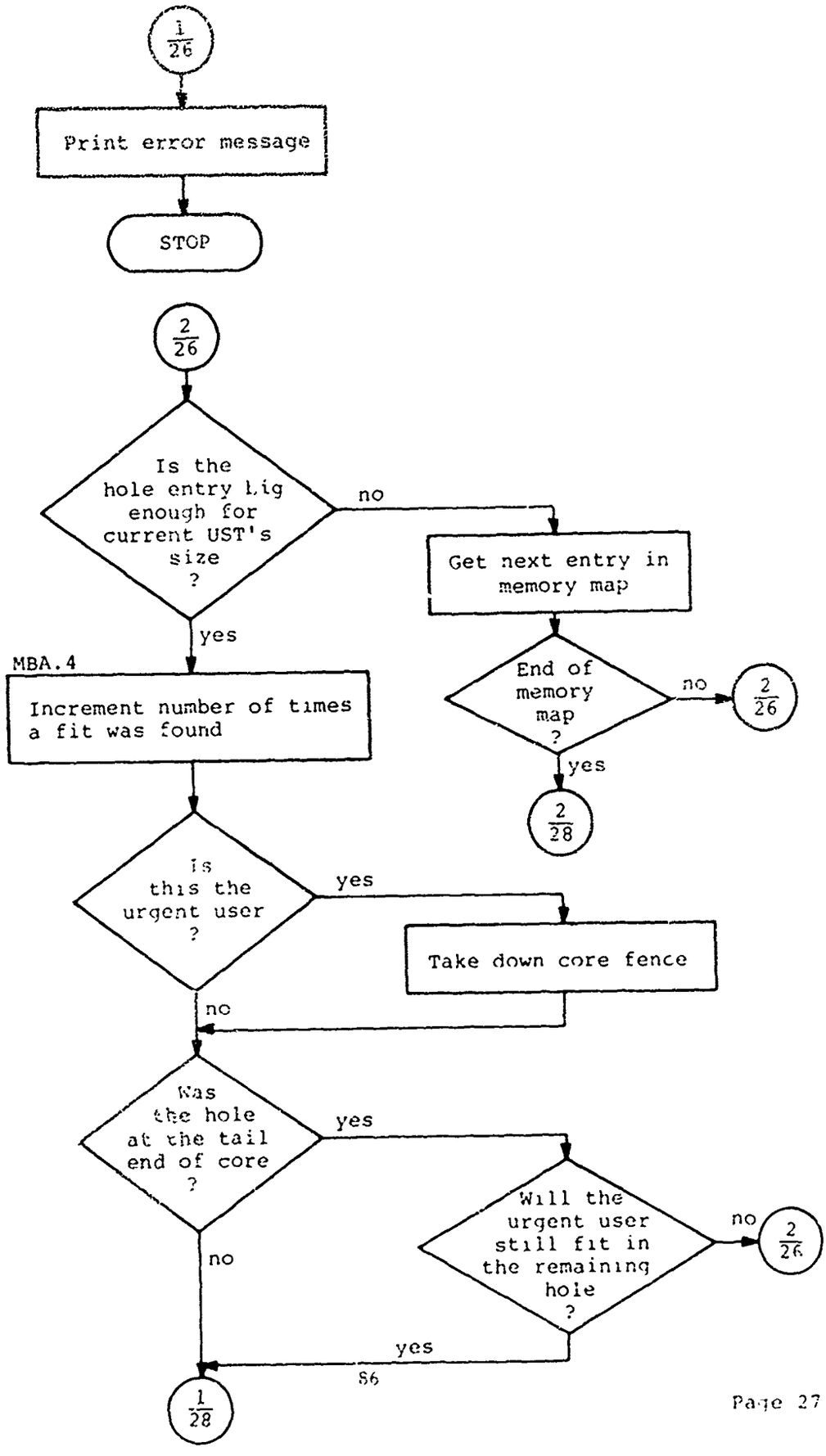

SDP7
Call SWOUT to swap user out
Increment count of force swaps for system and this UST
Move UST from CPUQ to MEMQ
ALLOCI

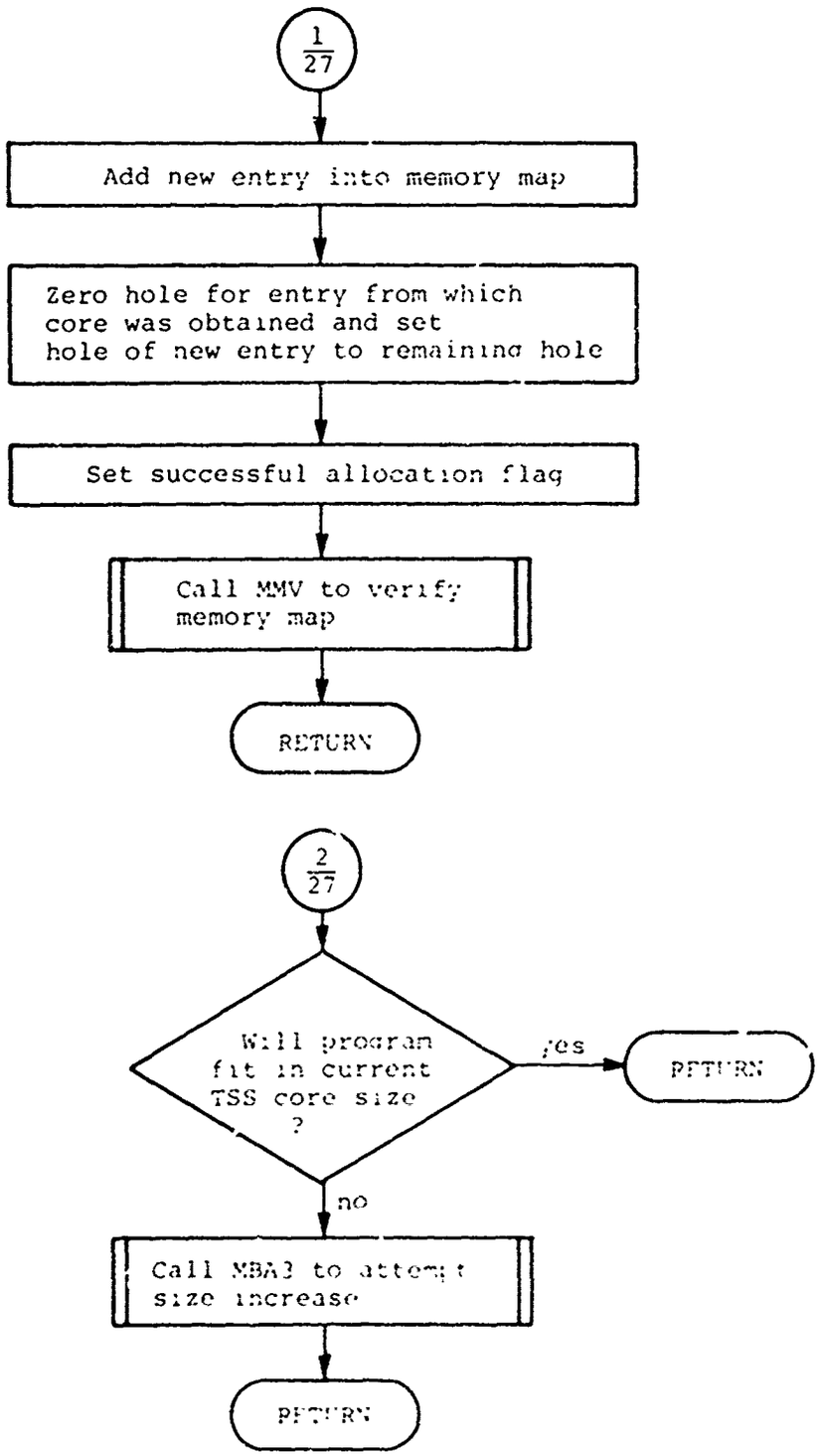

SPMACT
Size change waiting ?
no
Clear special memory action flag
RETURN
yes
Size decrease requested ?
yes
no
Has it been more than TAGMI ms. since last size increase ?
no
RETURN
yes
Increment count of size increases
1/25
SPM.3
Is new size requested equal to current size ?
yes
2/25
no
1/24

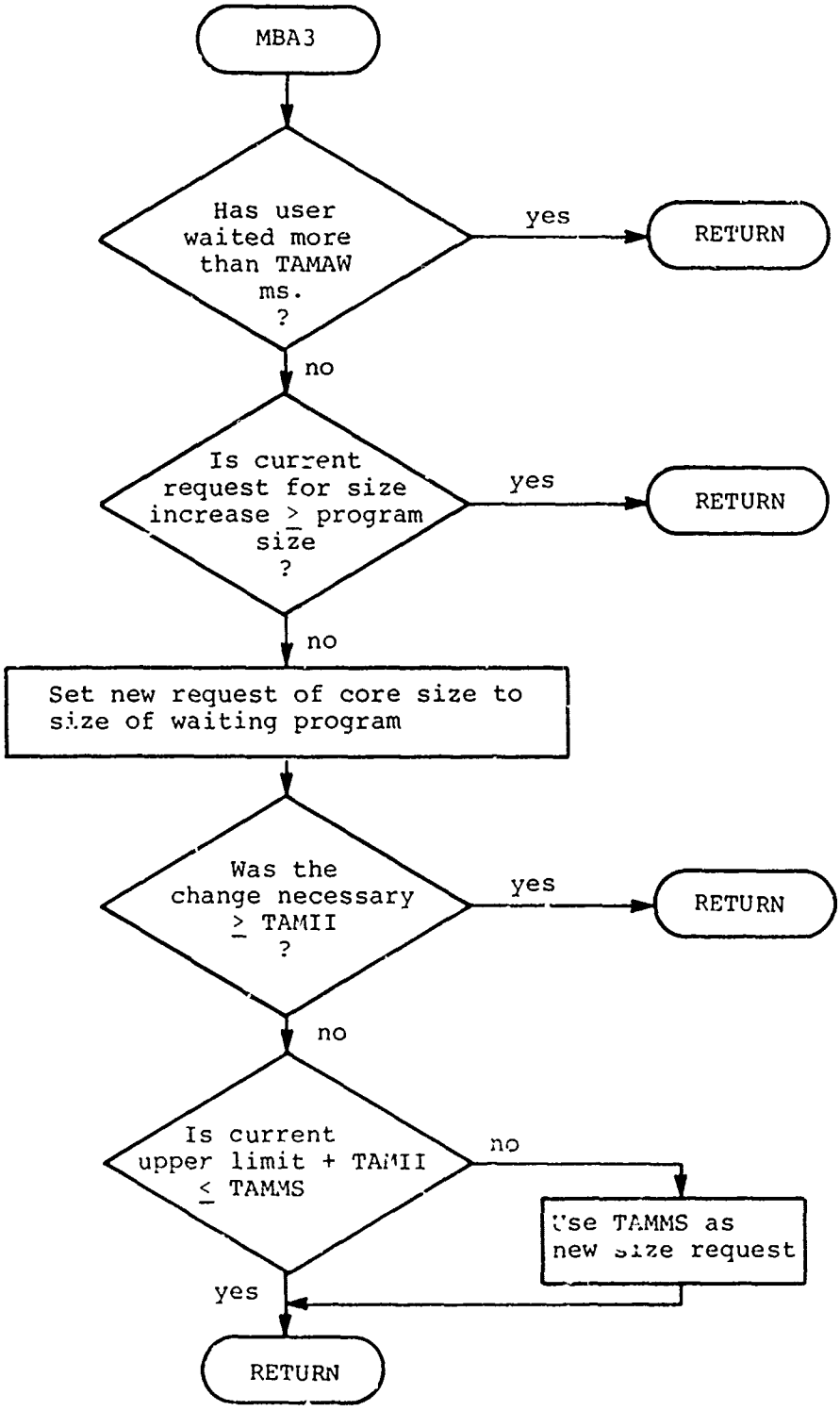

1/23
Is new size requested ± 3K ?
yes
2/25
no
Is hole size of last map entry big enough for reduction ?
no
4/25
yes
Has request been pending more than TATMC ms. ?
yes
4/25
no
Has request been pending less than TATMD ms. ?
yes
4/25
no
Increment count of size re uctions
1/25

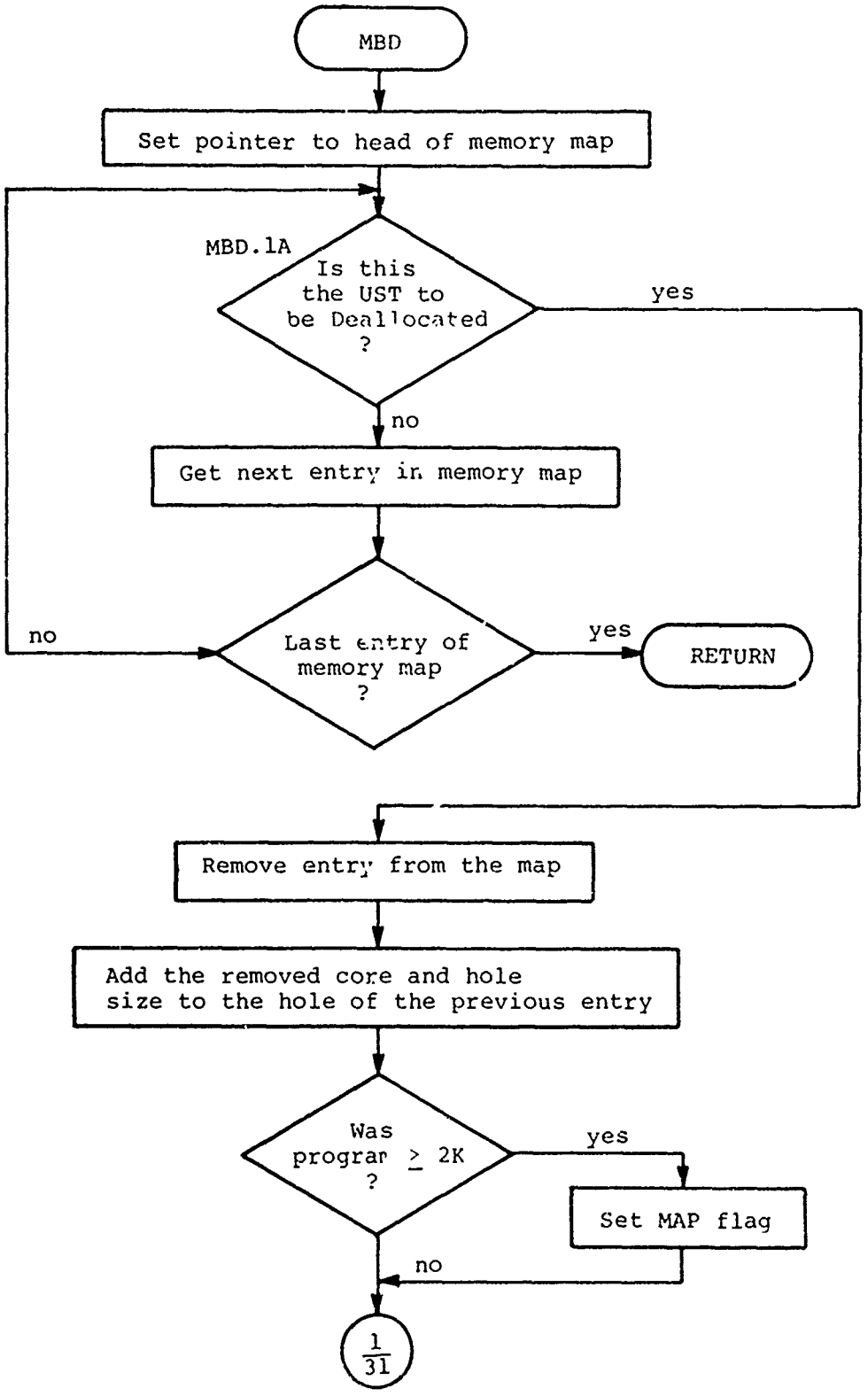

1
23
24
SPM.4
Update hole size of last entry to accomplish either size change
2
23
24
26
SPM.4A
Clear size change request
Clear special memory action flag and time of reduction request refusal
3
26
SPM.4B
Set time of last size change
RETURN
4
24
Has change been refused before ?
no
Set time of change refusal
yes
1
26

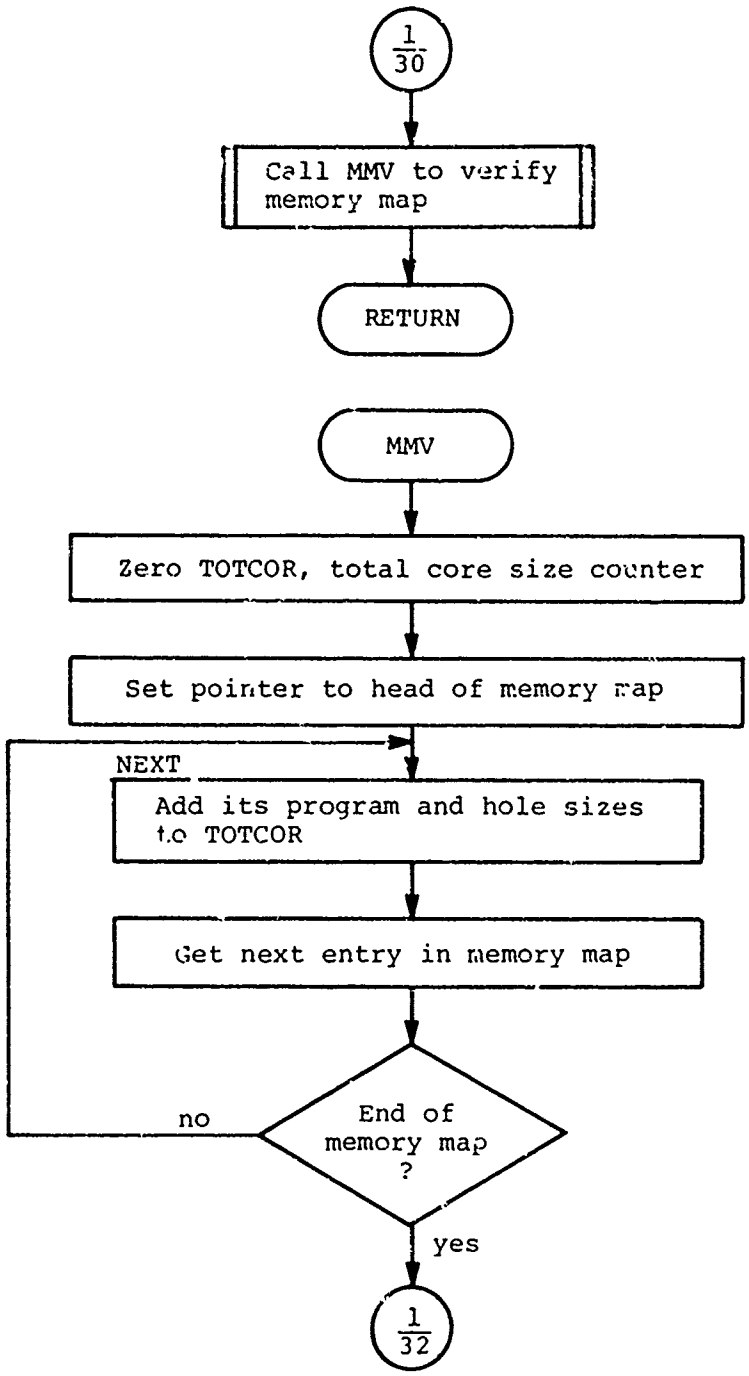

1/25
Is time since first refusal ≥ TATMC ms. ?
yes
2/25
no
3/25
MBA
Clear success flag
Increment count of entries to MBA
Set pointer to head of memory map
2/27
no
MBA.1
Is this UST same as current UST ?
yes
1/27
no
2/27

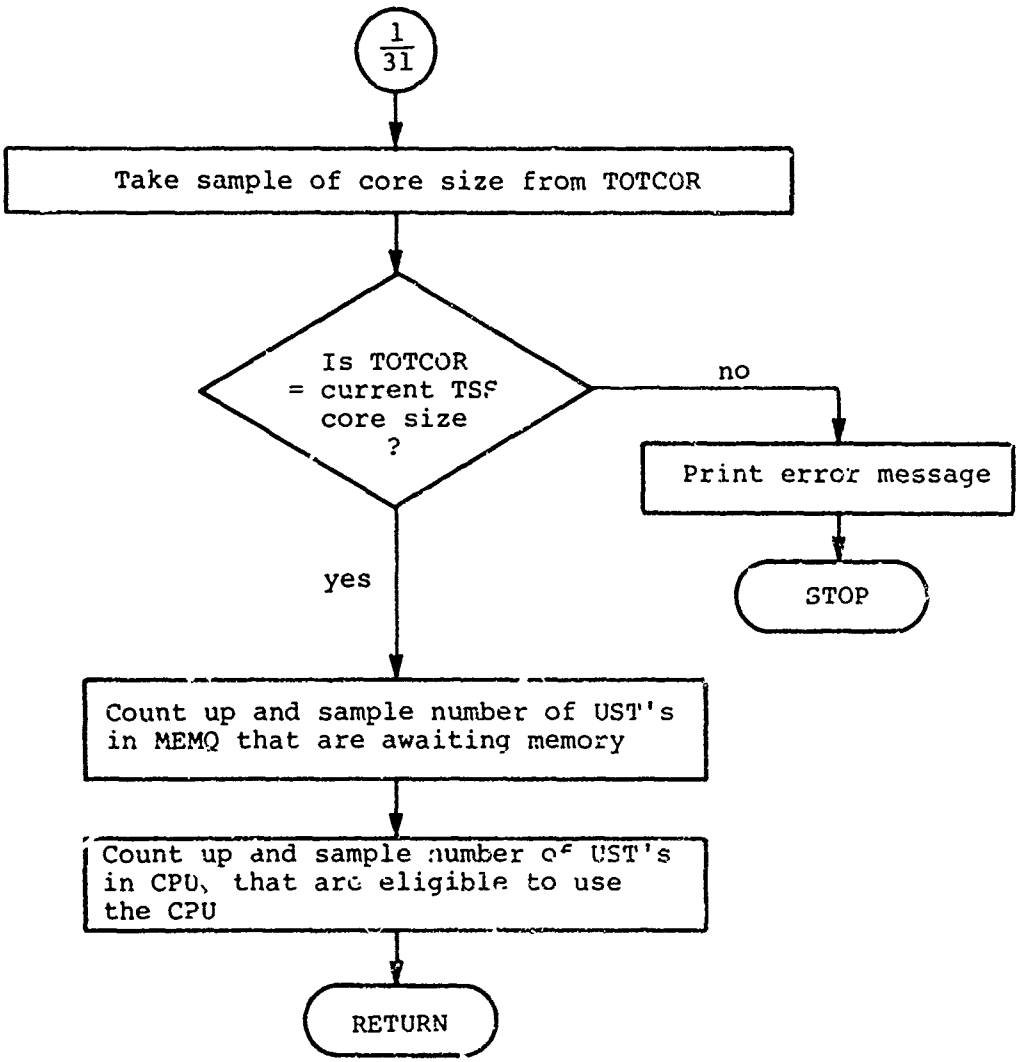

1/26
Print error message
STOP
2/26
Is the hole entry big enough for current UST's size ?
no
Get next entry in memory map
End of memory map ?
no
2/26
yes
2/28
yes
MBA.4
Increment number of times a fit was found
Is this the urgent user ?
yes
Take down core fence
no
Was the hole at the tail end of core ?
yes
Will the urgent user still fit in the remaining hole ?
no
2/26
no
yes
1/28

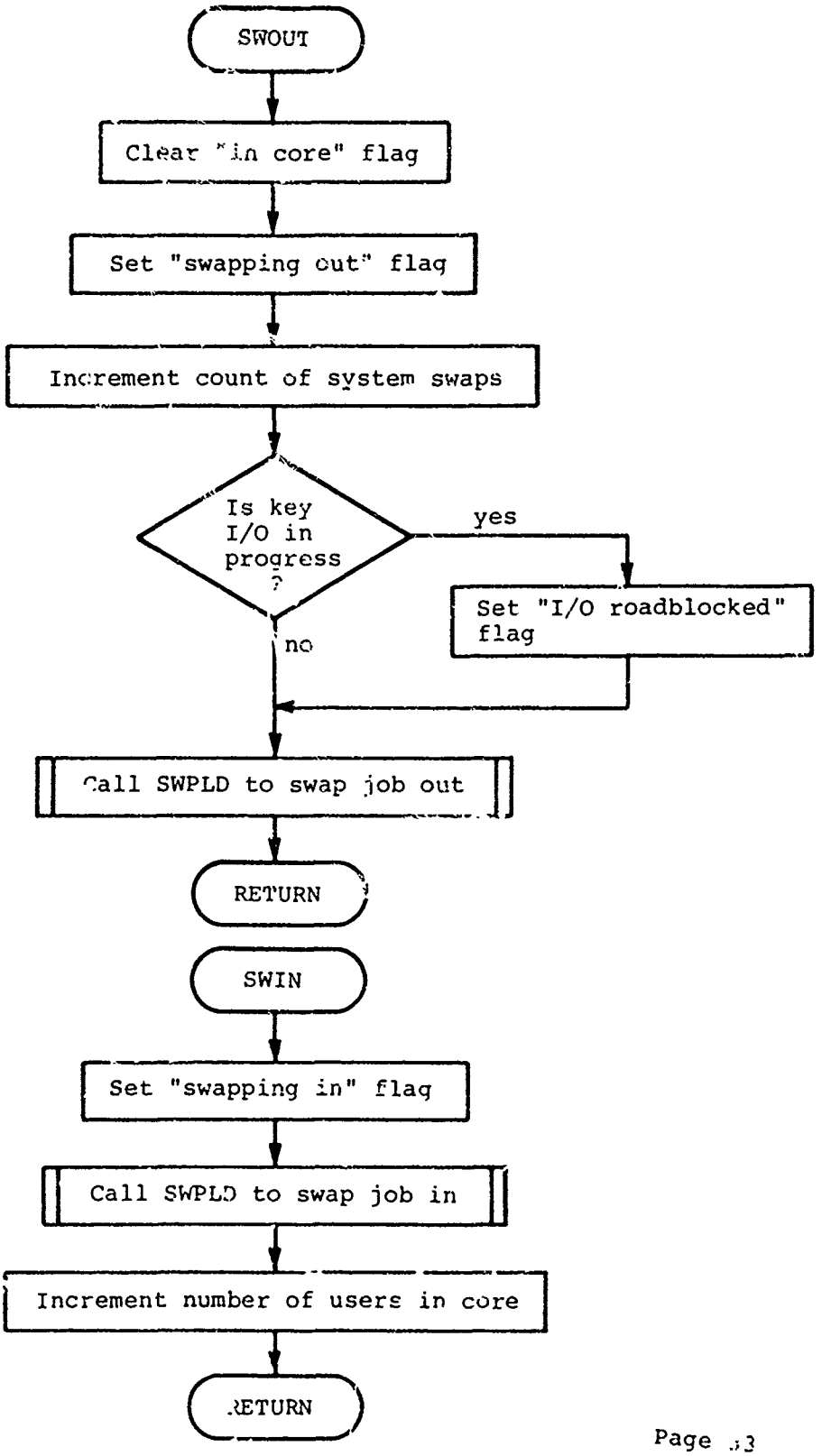

1/27
Add new entry into memory map
Zero hole for entry from which core was obtained and set hole of new entry to remaining hole
Set successful allocation flag
Call MMV to verify memory map
RETURN
2/27
Will program fit in current TSS core size ?
yes
RETURN
no
Call MBA3 to attempt size increase
RETURN


MBA3
Has user waited more than TAMAW ms. ?
yes
RETURN
no
Is current request for size increase ≥ program size ?
yes
RETURN
no
Set new request of core size to size of waiting program
Was the change necessary ≥ TAMII ?
yes
RETURN
no
Is current upper limit + TAMII ≤ TAMMS
no
Use TAMMS as new size request
yes
RETURN

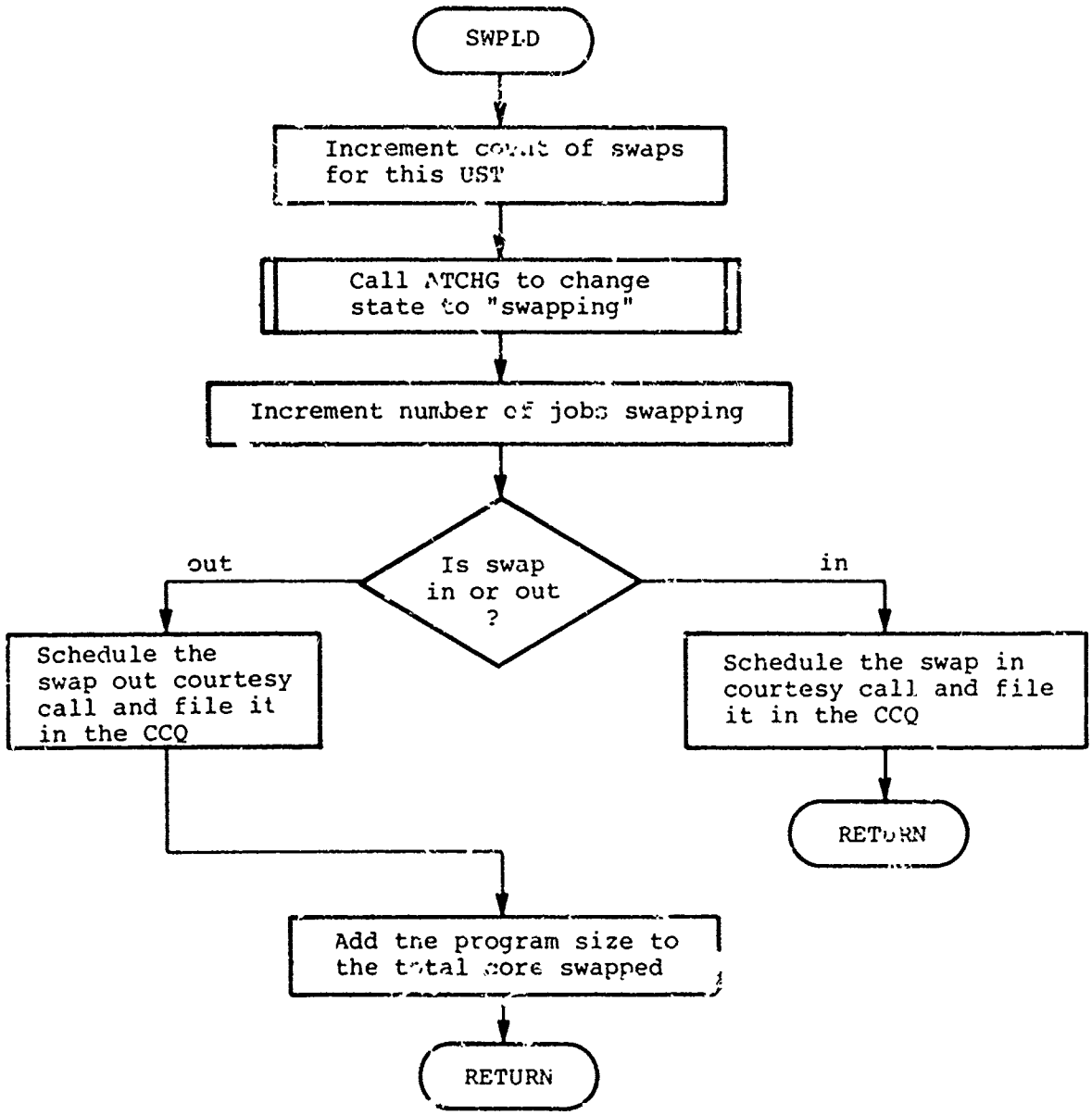

MBD
Set pointer to head of memory map
MBD.1A
Is this the UST to be Deallocated ?
yes
no
Get next entry in memory map
Last entry of memory map ?
no
yes
RETURN
Remove entry from the map
Add the removed core and hole size to the hole of the previous entry
Was progra ≥ 2K ?
yes
Set MAP flag
no
1/31

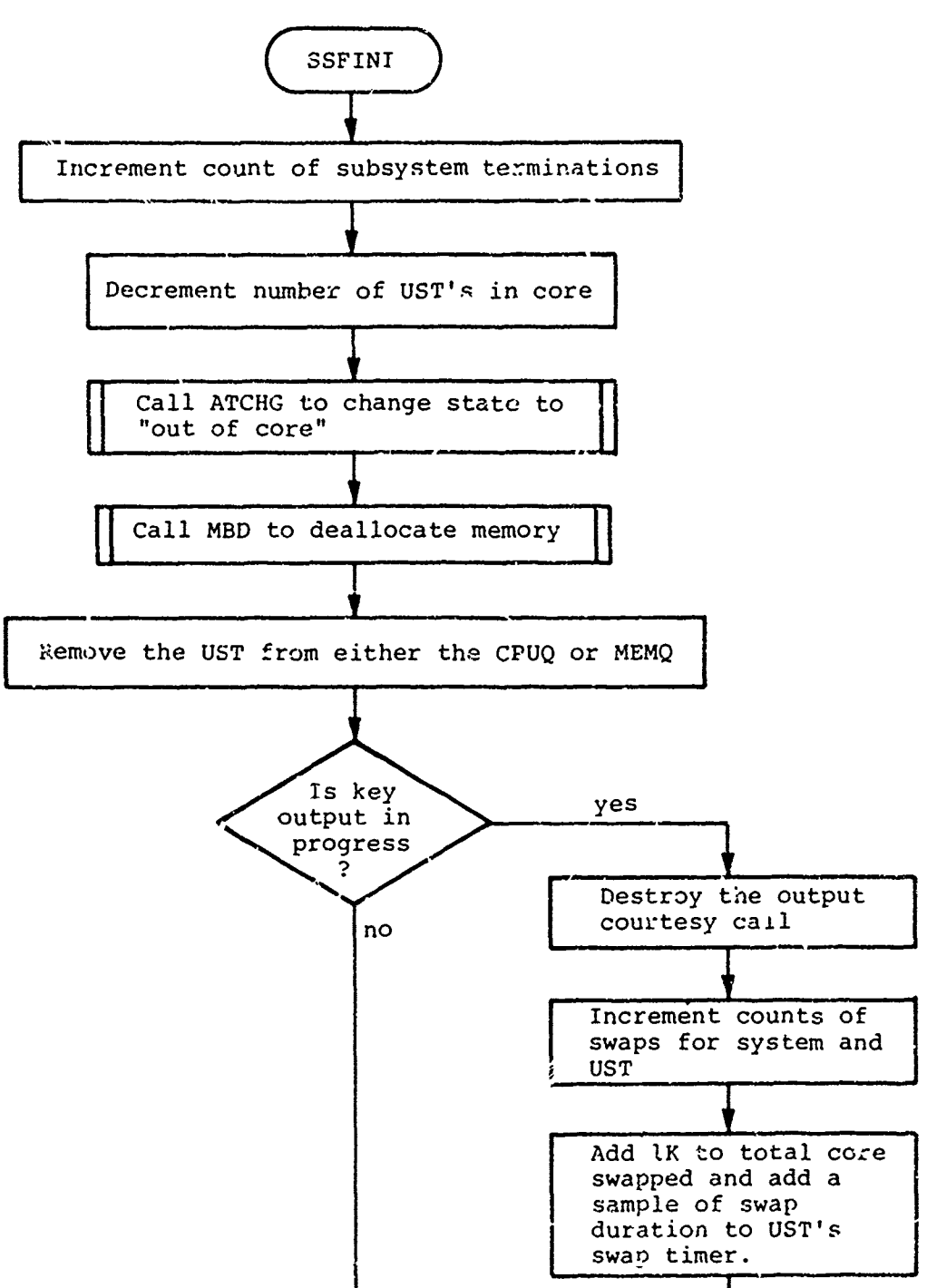

1/30
Call MMV to verify memory map
RETURN
MMV
Zero TOTCOR, total core size counter
Set pointer to head of memory map
NEXT
Add its program and hole sizes to TOTCOR
Get next entry in memory map
End of memory map ?
no
yes
1/32

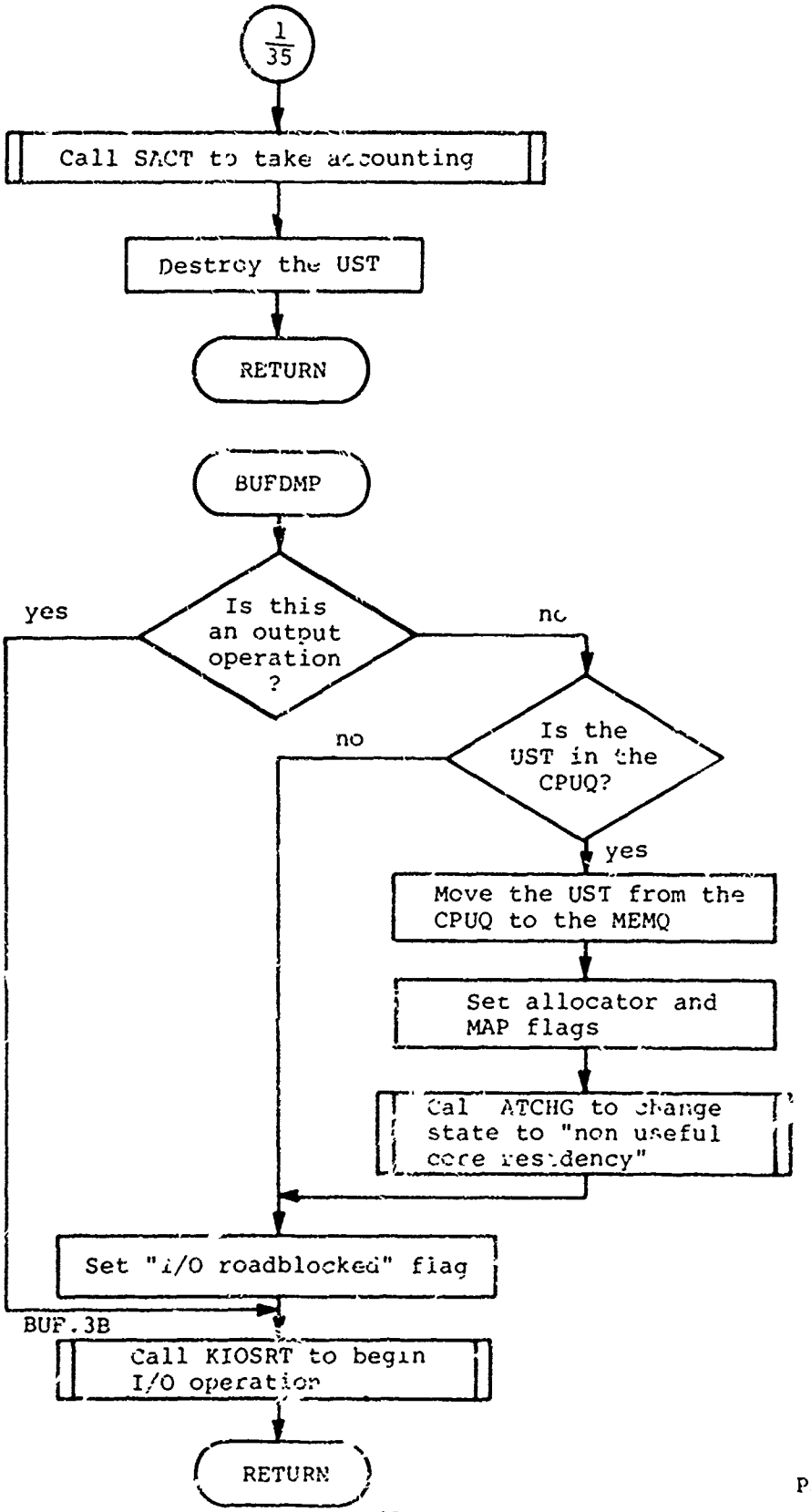

1/31
Take sample of core size from TOTCOR
Is TOTCOR = current TSF core size ?
no
Print error message
STOP
yes
Count up and sample number of UST's in MEMQ that are awaiting memory
Count up and sample number of UST's in CPU, that are eligible to use the CPU
RETURN

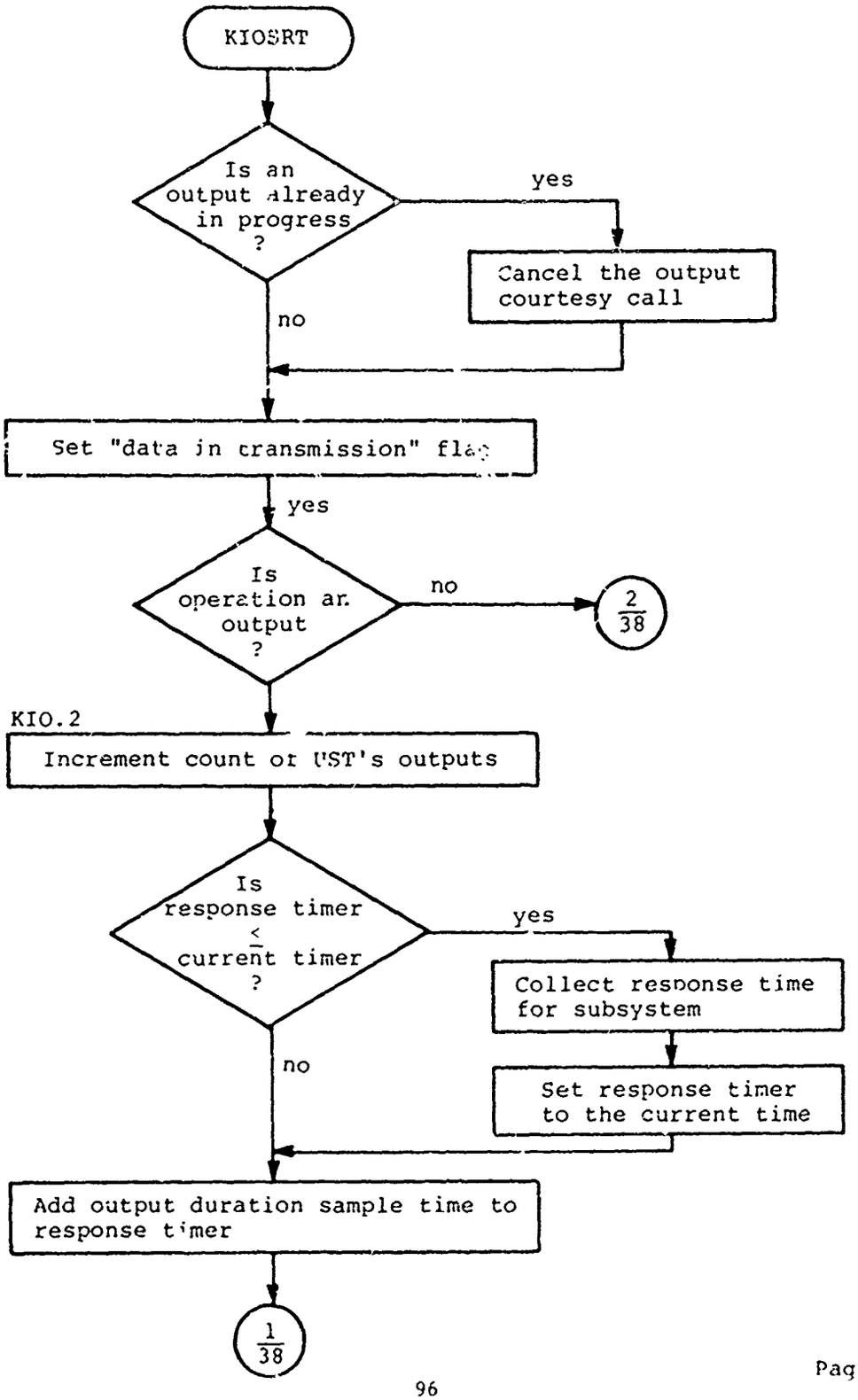

SWOUT
Clear "in core" flag
Set "swapping out" flag
Increment count of system swaps
Is key I/O in progress ?
yes
Set "I/O roadblocked" flag
no
Call SWPLD to swap job out
RETURN
SWIN
Set "swapping in" flag
Call SWPLD to swap job in
Increment number of users in core
RETURN

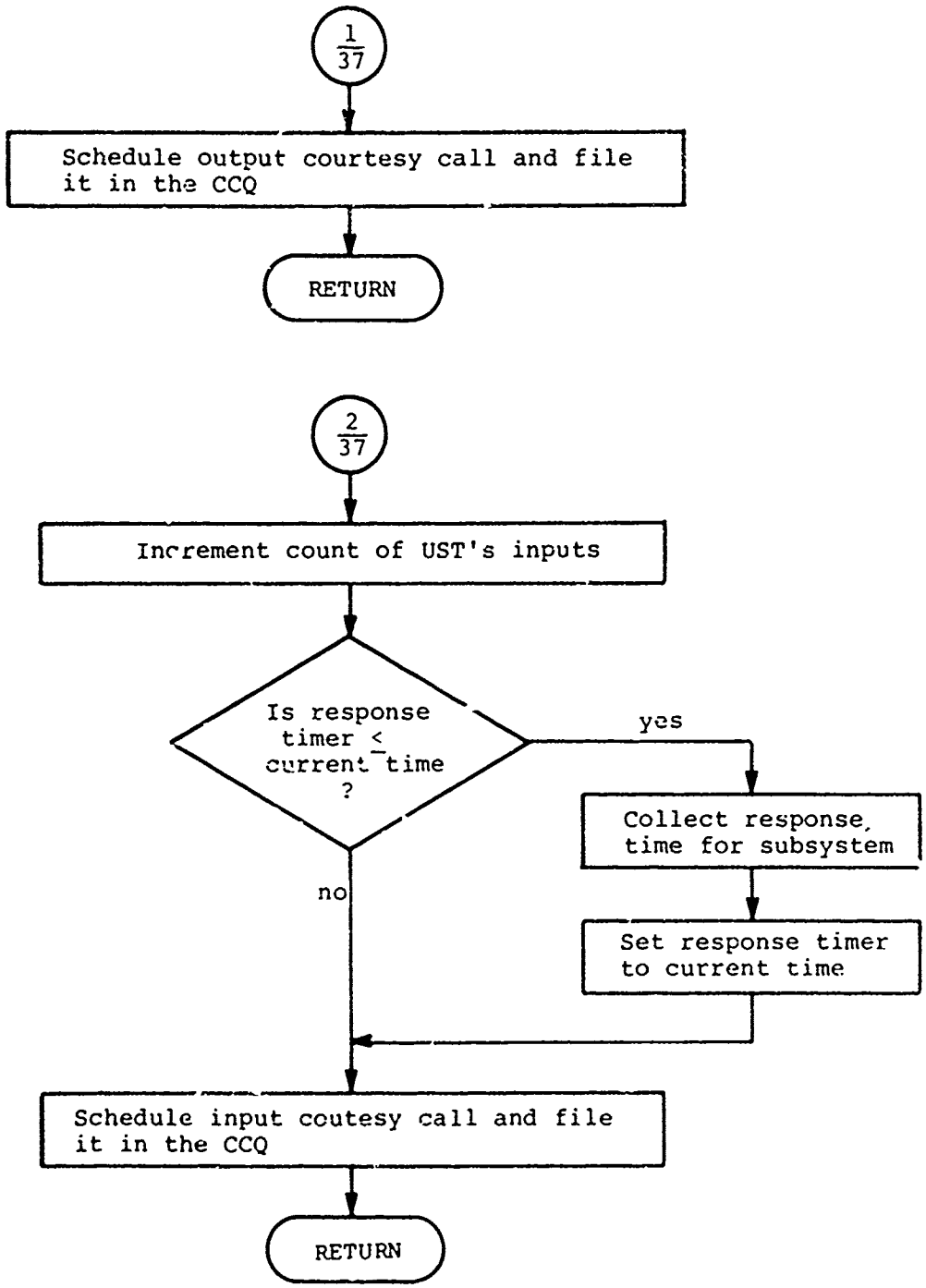
Page 93

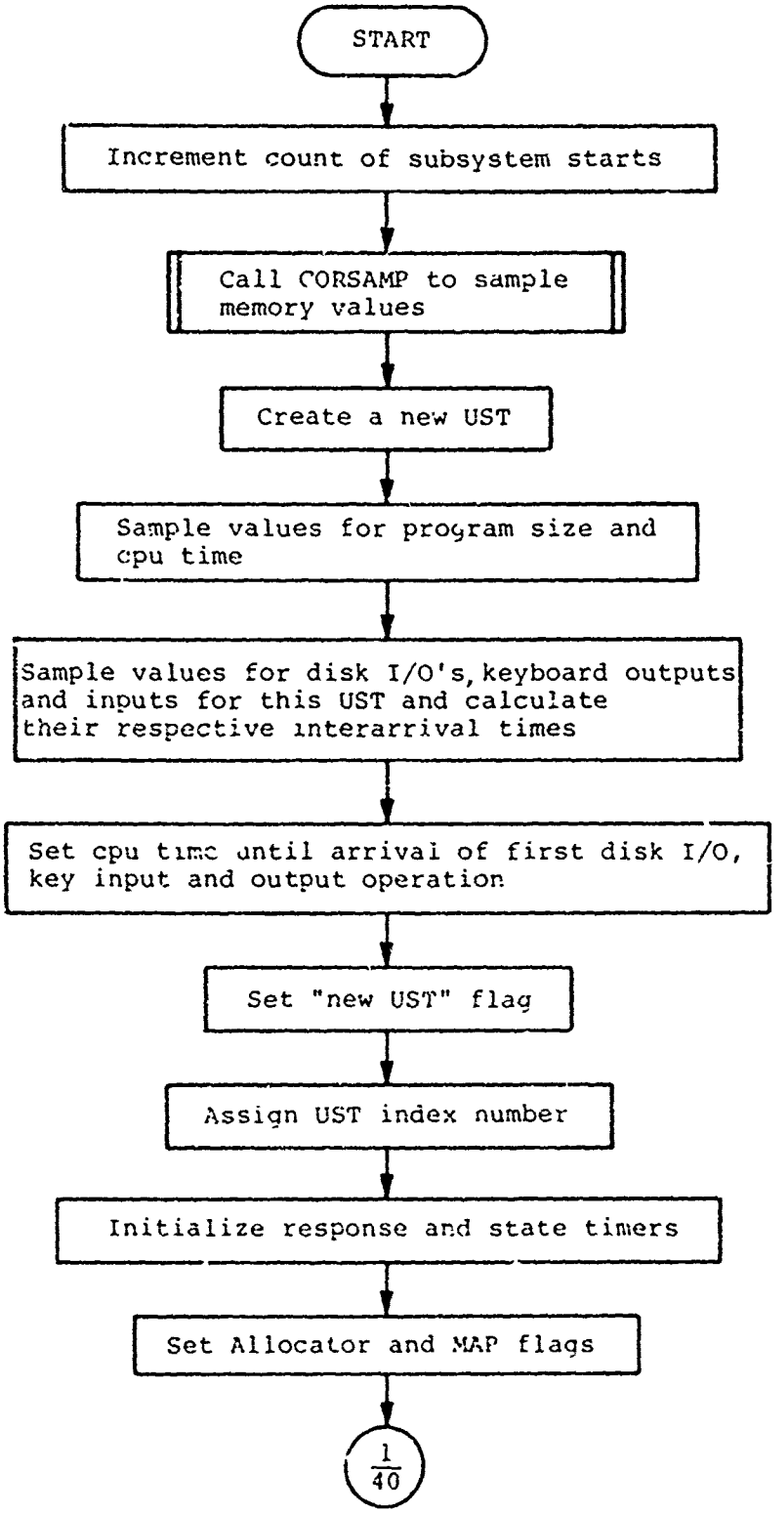

SWPLD
Increment count of swaps for this UST
Call ATCHG to change state to "swapping"
Increment number of jobs swapping
Is swap in or out ?
out
in
Schedule the swap out courtesy call and file it in the CCQ
Schedule the swap in courtesy call and file it in the CCQ
RETURN
Add the program size to the total core swapped
RETURN

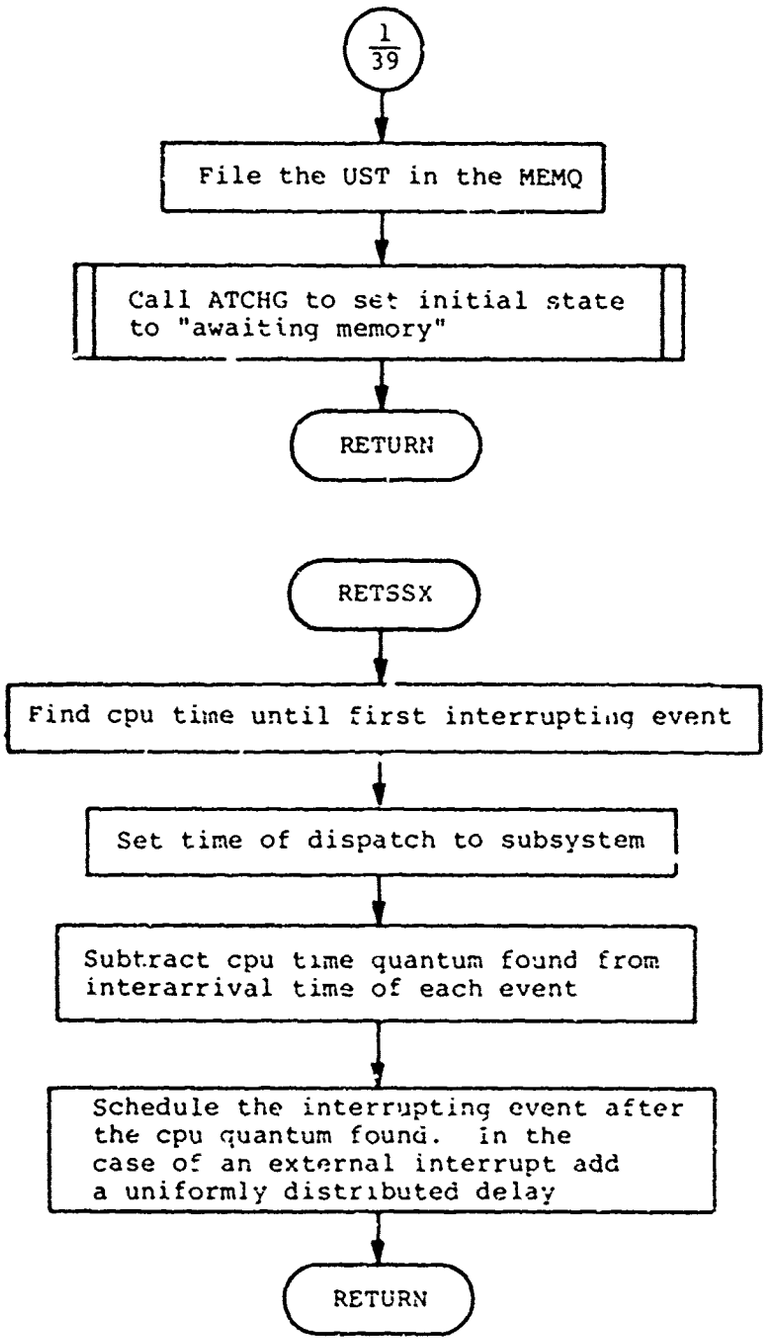

SSFINI
Increment count of subsystem terminations
Decrement number of UST's in core
Call ATCHG to change state to "out of core"
Call MBD to deallocate memory
Remove the UST from either the CPUQ or MEMQ
Is key output in progress ?
yes
no
Destroy the output courtesy call
Increment counts of swaps for system and UST
Add 1K to total core swapped and add a sample of swap duration to UST's swap timer.
1 36

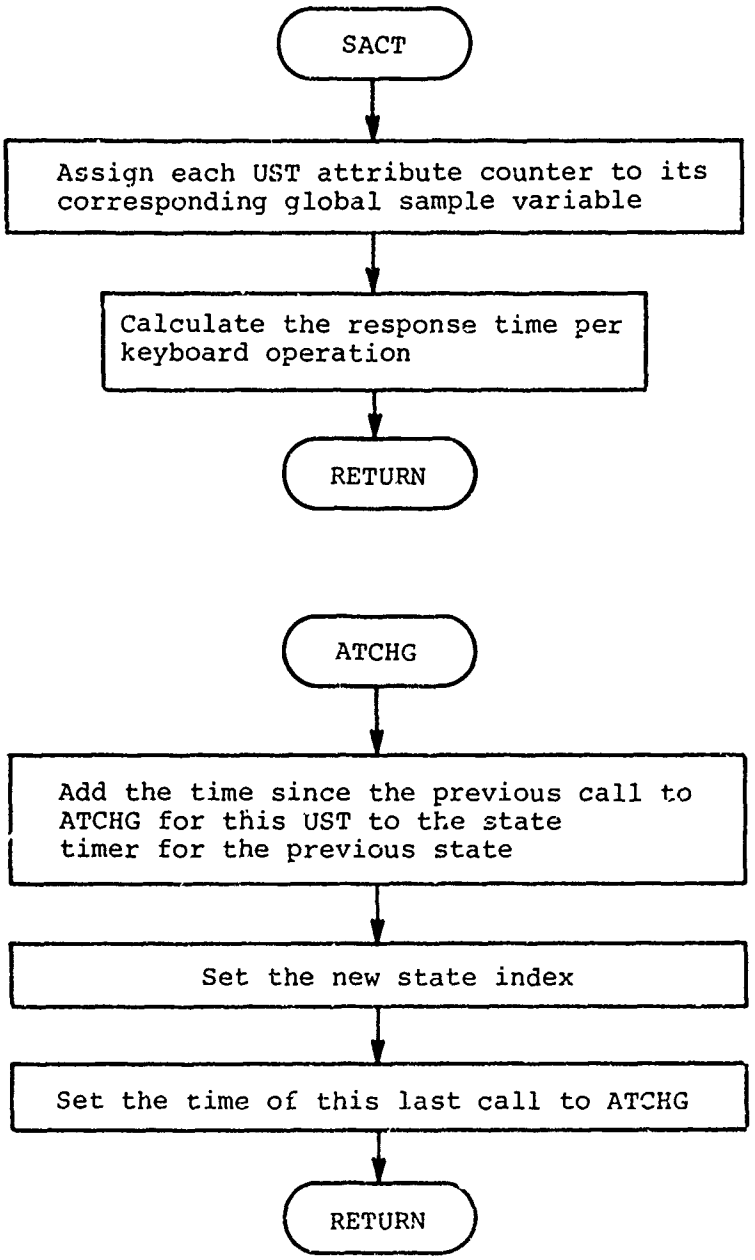

1/35
Call SACT to take accounting
Destroy the UST
RETURN
BUFDMP
Is this an output operation ?
yes
no
Is the UST in the CPUQ?
no
yes
Move the UST from the CPUQ to the MEMQ
Set allocator and MAP flags
Cal ATCHG to change state to "non useful core residency"
Set "I/O roadblocked" flag
BUF.3B
Call KIOSRT to begin I/O operation
RETURN

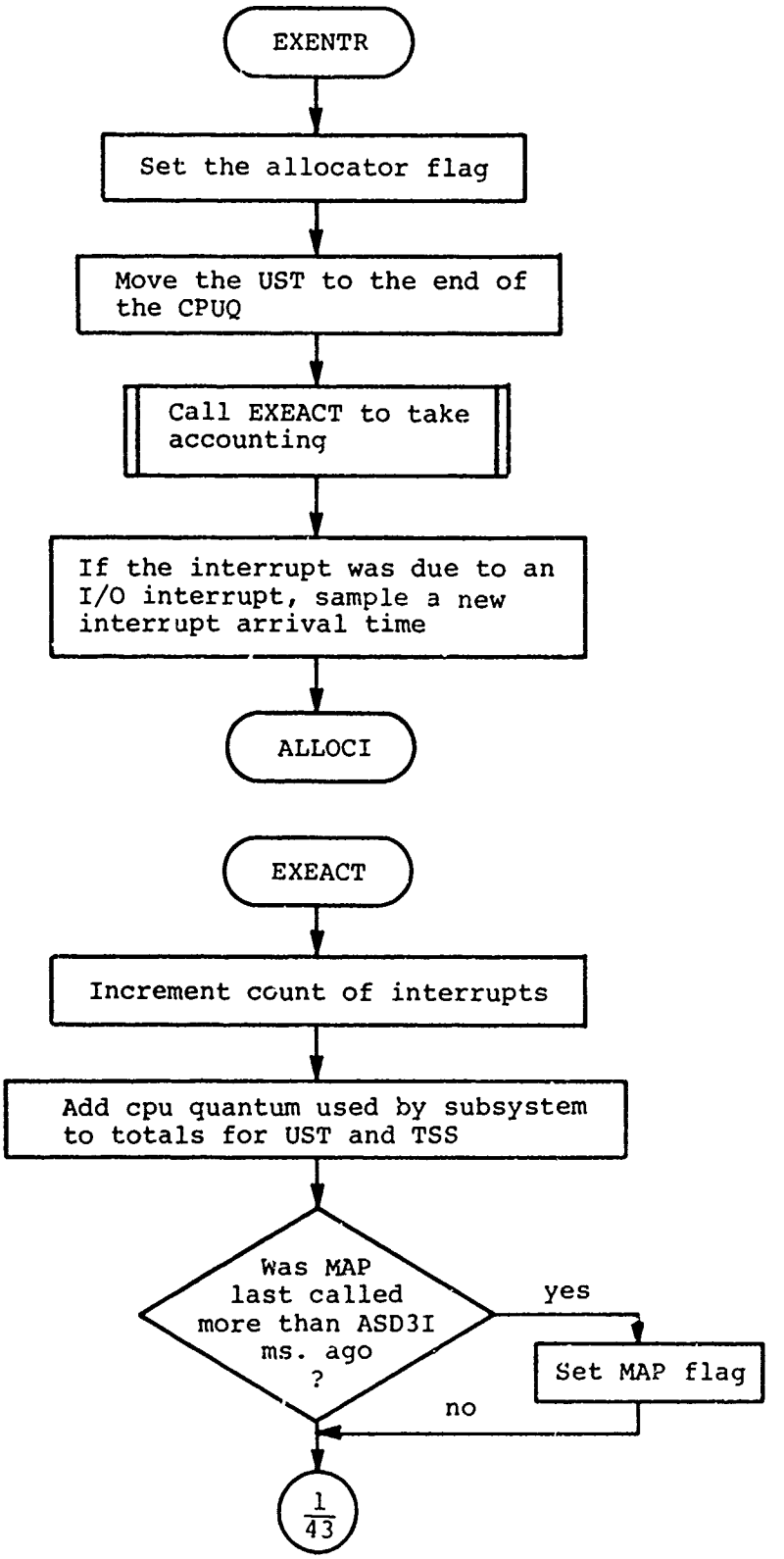

KIOSRT
Is an output already in progress ?
yes
Cancel the output courtesy call
no
Set "data in transmission" flag
yes
Is operation an output ?
no
2/38
KIO.2
Increment count of UST's outputs
Is response timer < current timer ?
yes
Collect response time for subsystem
Set response timer to the current time
no
Add output duration sample time to response timer
1/38

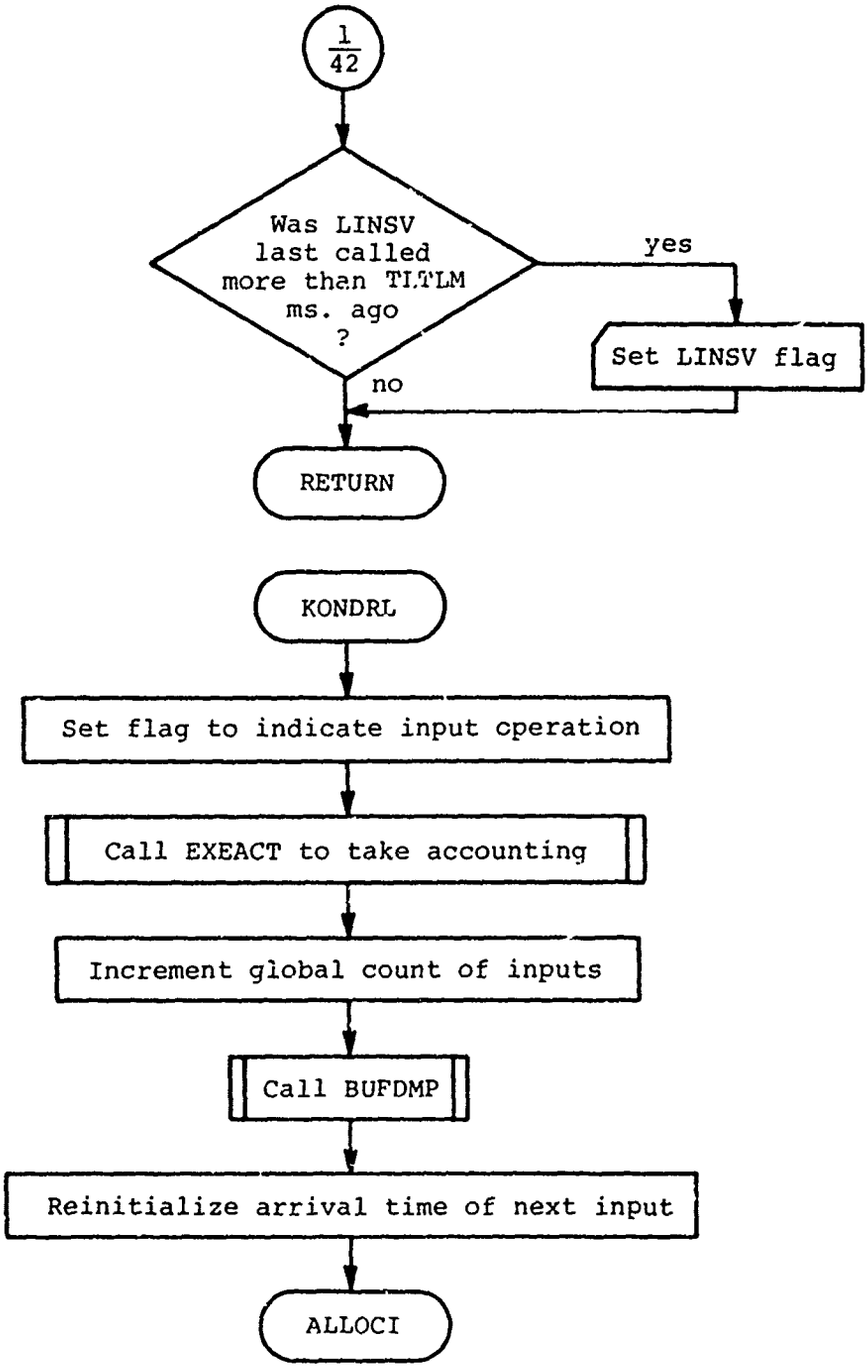

1/37
Schedule output courtesy call and file it in the CCQ
RETURN
2/37
Increment count of UST's inputs
Is response timer < current time ?
yes
Collect response time for subsystem
Set response timer to current time
no
Schedule input coutesy call and file it in the CCQ
RETURN

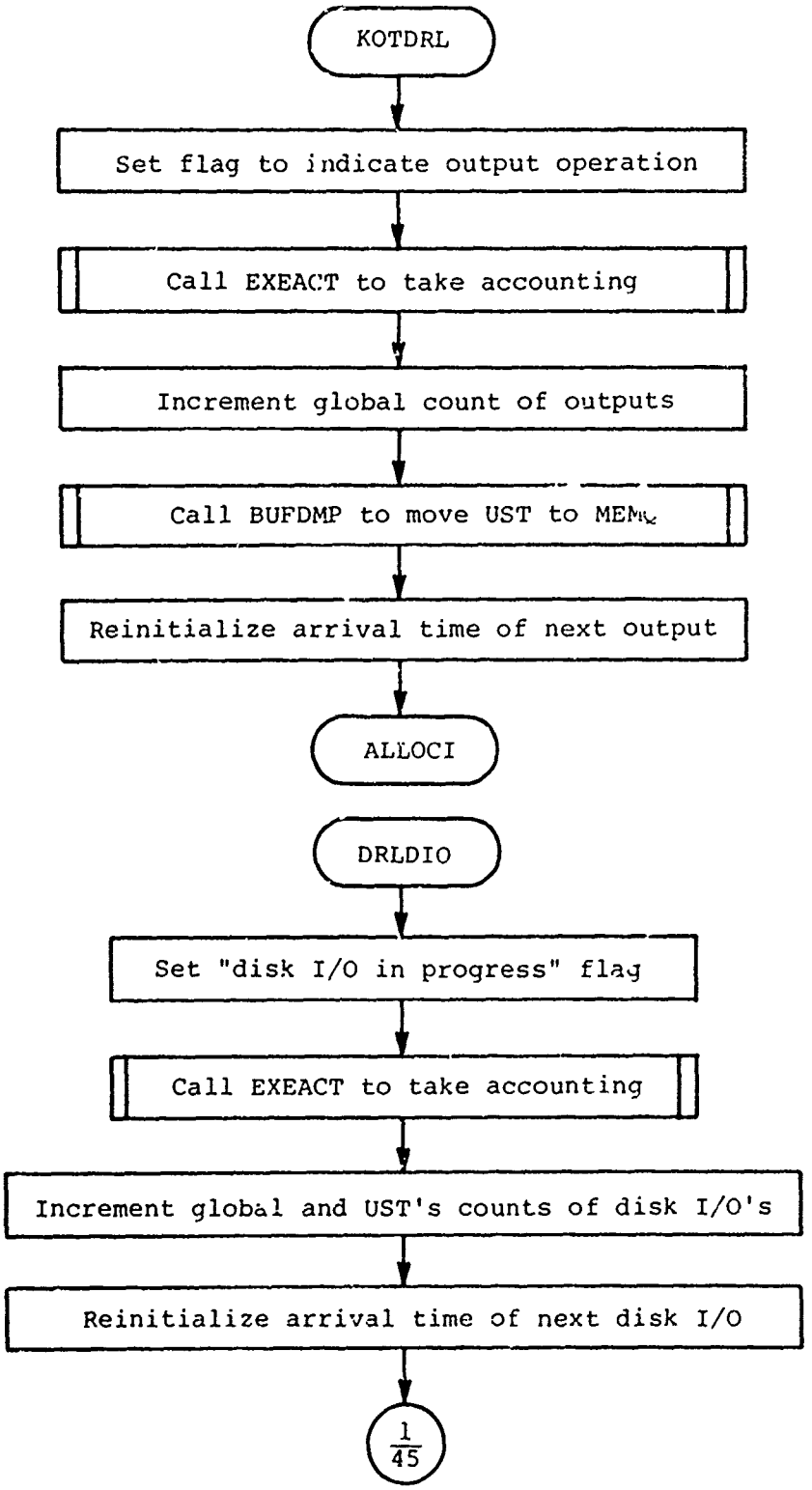

START
Increment count of subsystem starts
Call CORSAMP to sample memory values
Create a new UST
Sample values for program size and cpu time
Sample values for disk I/O's, keyboard outputs and inputs for this UST and calculate their respective interarrival times
Set cpu time until arrival of first disk I/O, key input and output operation
Set "new UST" flag
Assign UST index number
Initialize response and state timers
Set Allocator and MAP flags
1/40

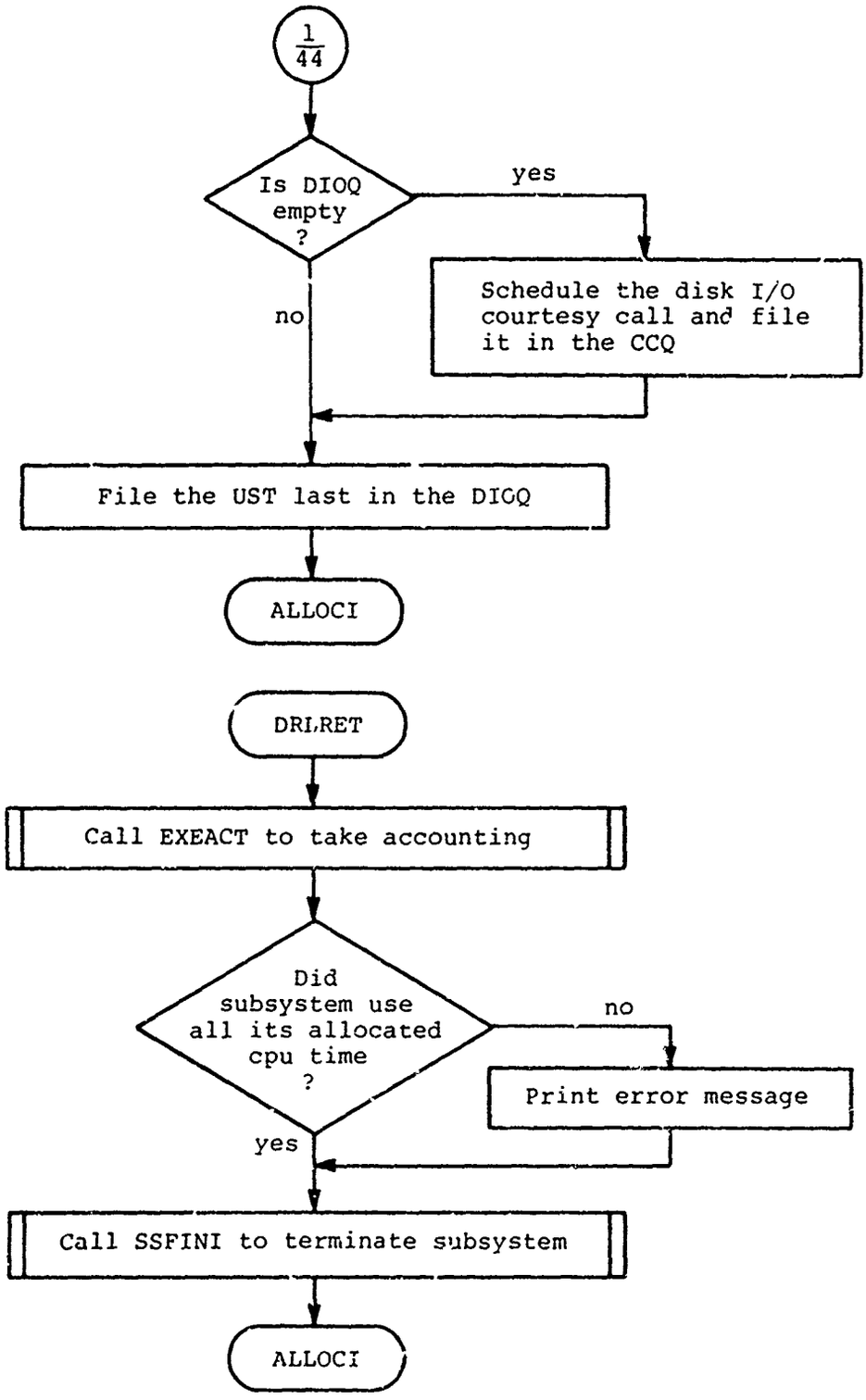

1/39
File the UST in the MEMQ
Call ATCHG to set initial state to "awaiting memory"
RETURN
RETSSX
Find cpu time until first interrupting event
Set time of dispatch to subsystem
Subtract cpu time quantum found from interarrival time of each event
Schedule the interrupting event after the cpu quantum found. in the case of an external interrupt add a uniformly distributed delay
RETURN

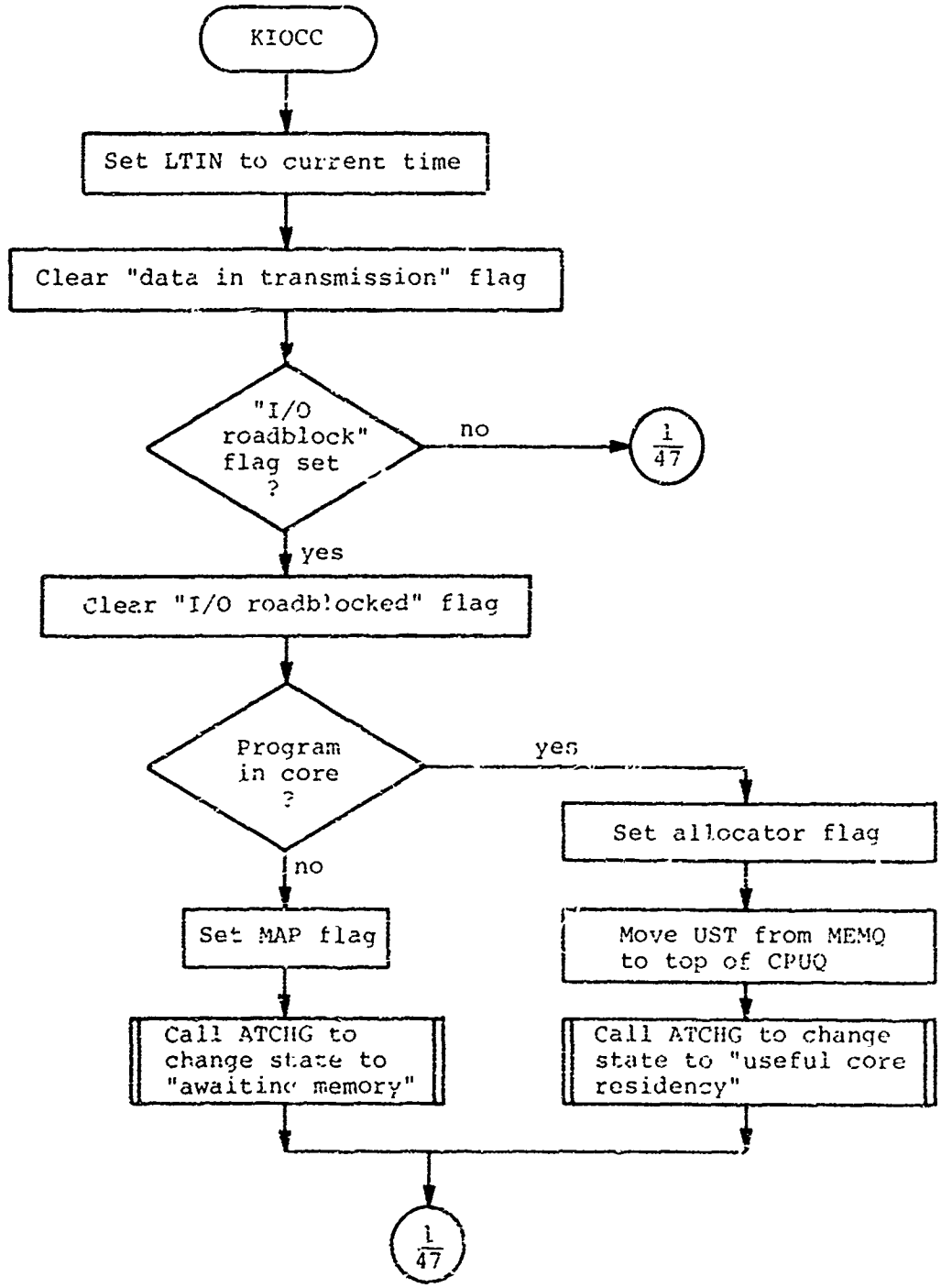

SACT
Assign each UST attribute counter to its corresponding global sample variable
Calculate the response time per keyboard operation
RETURN
ATCHG
Add the time since the previous call to ATCHG for this UST to the state timer for the previous state
Set the new state index
Set the time of this last call to ATCHG
RETURN

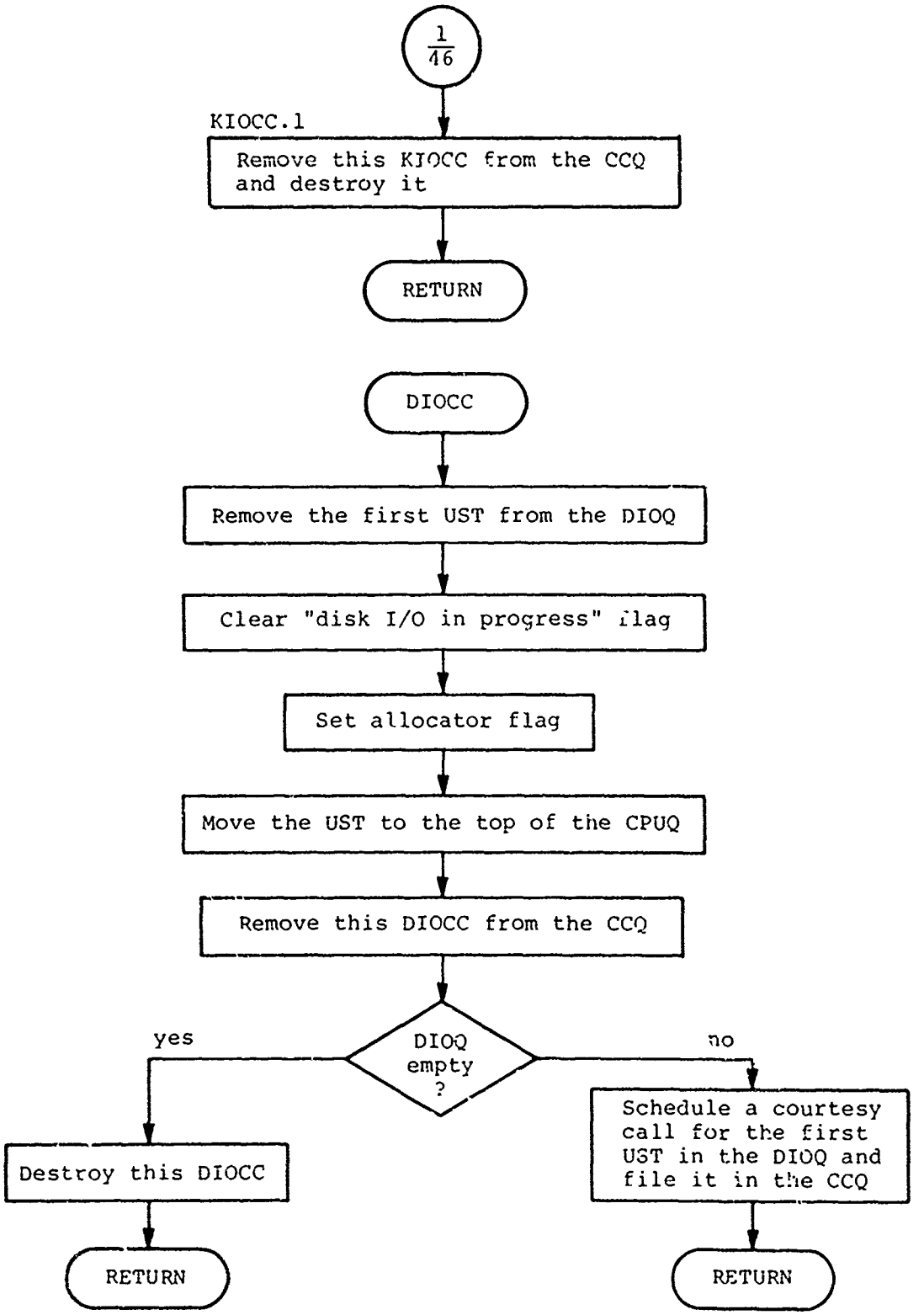

EXENTR
Set the allocator flag
Move the UST to the end of the CPUQ
Call EXEACT to take accounting
If the interrupt was due to an I/O interrupt, sample a new interrupt arrival time
ALLOCI
EXEACT
Increment count of interrupts
Add cpu quantum used by subsystem to totals for UST and TSS
Was MAP last called more than ASD3I ms. ago ?
yes
Set MAP flag
no
1/43

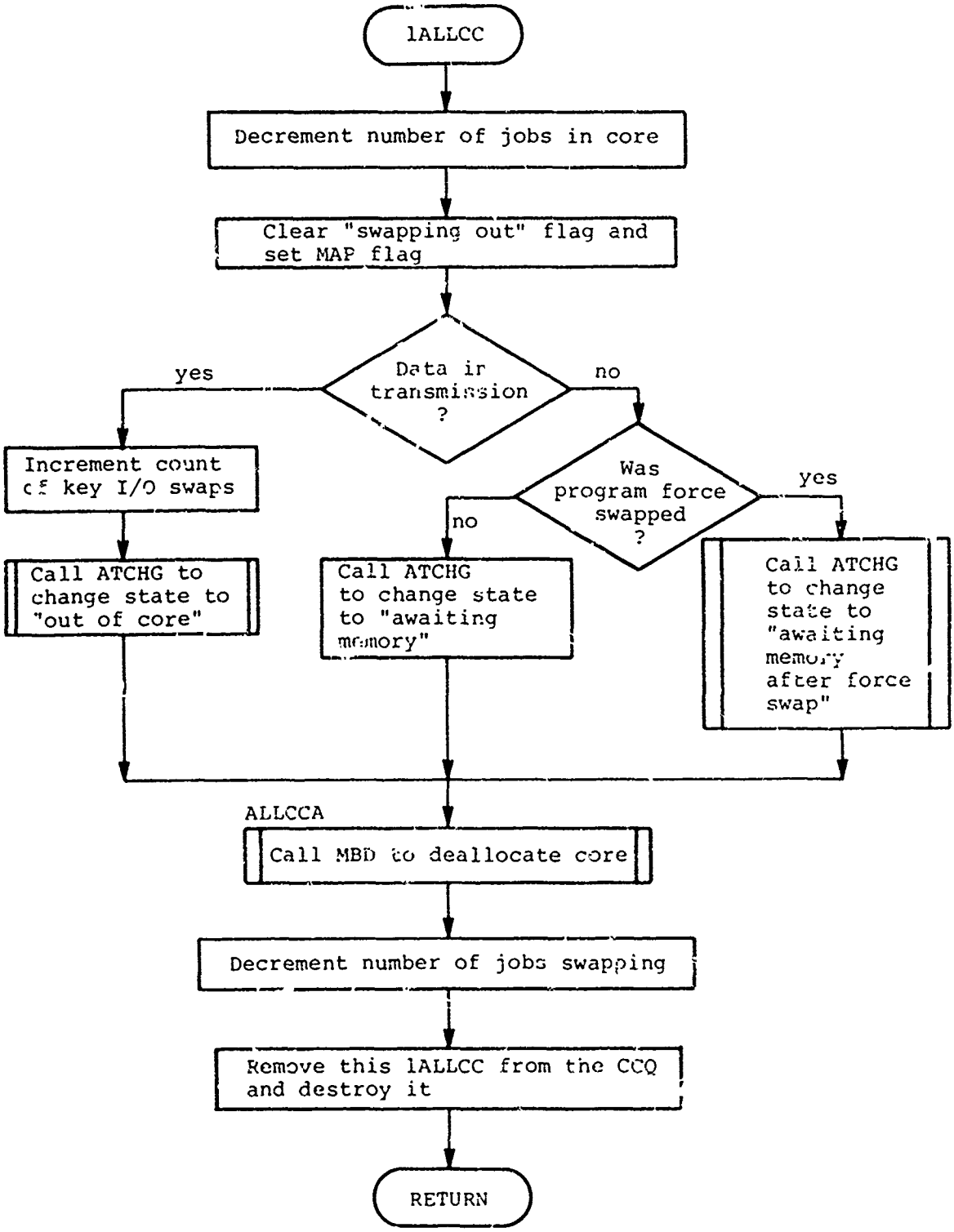

1/42
Was LINSV last called more than TLTLM ms. ago ?
yes
Set LINSV flag
no
RETURN
KONDRL
Set flag to indicate input operation
Call EXEACT to take accounting
Increment global count of inputs
Call BUFDMP
Reinitialize arrival time of next input
ALLOCI

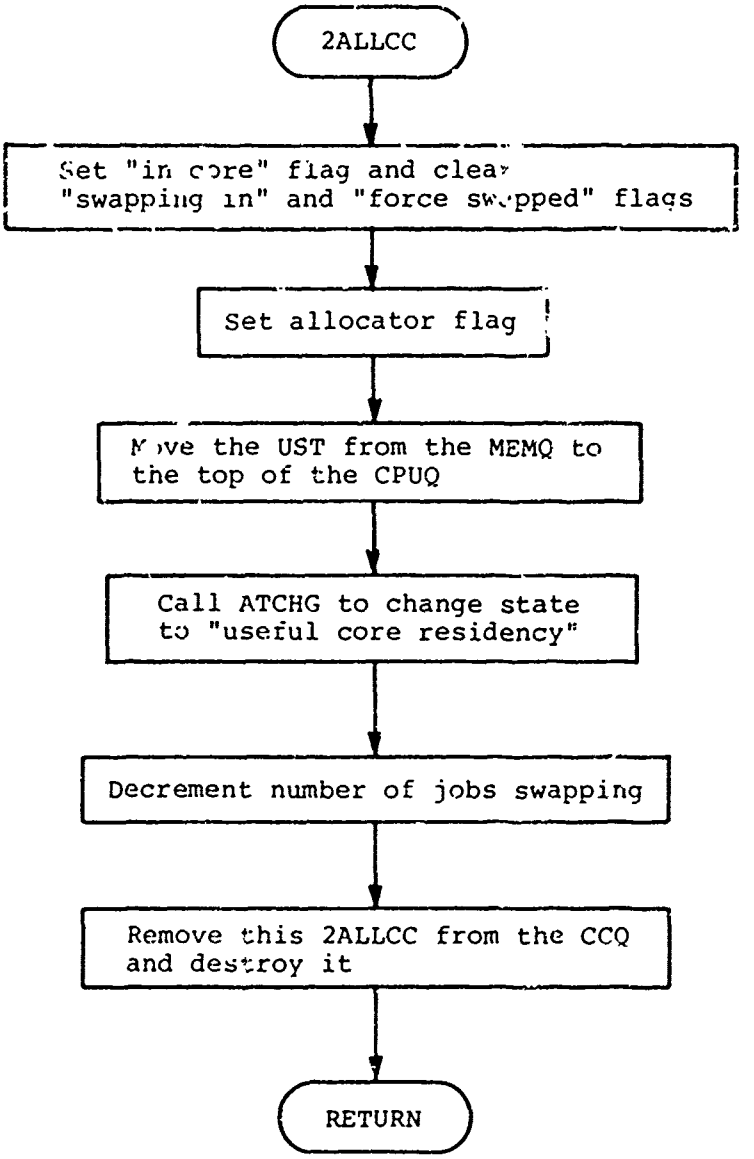

KOTDRL
Set flag to indicate output operation
Call EXEACT to take accounting
Increment global count of outputs
Call BUFDMP to move UST to MEM
Reinitialize arrival time of next output
ALLOCI
DRLDIO
Set "disk I/O in progress" flag
Call EXEACT to take accounting
Increment global and UST's counts of disk I/O's
Reinitialize arrival time of next disk I/O
1/45

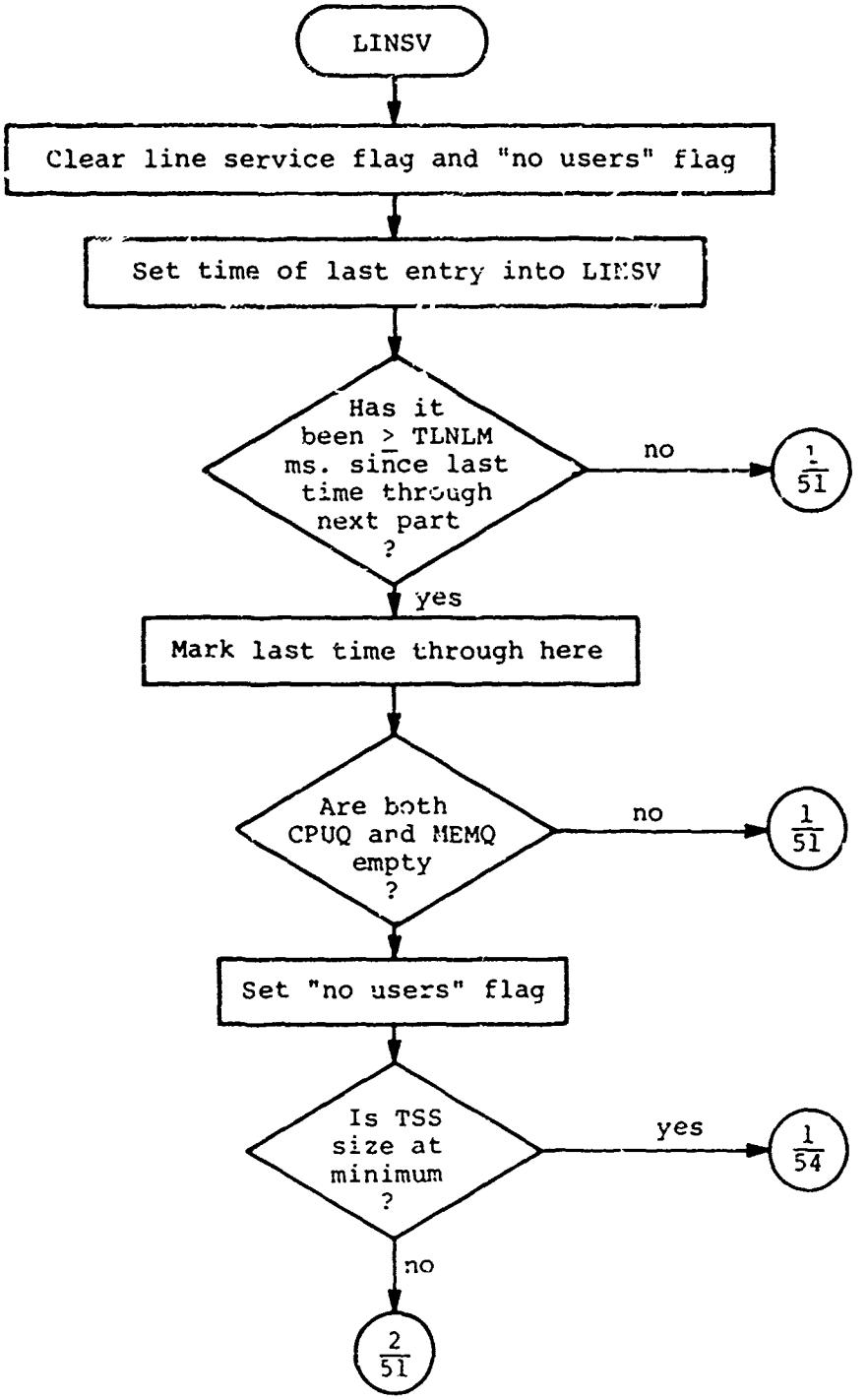

1/44
Is DIOQ empty ?
yes
no
Schedule the disk I/O courtesy call and file it in the CCQ
File the UST last in the DIOQ
ALLOCI
DRLRET
Call EXEACT to take accounting
Did subsystem use all its allocated cpu time ?
no
yes
Print error message
Call SSFINI to terminate subsystem
ALLOCI

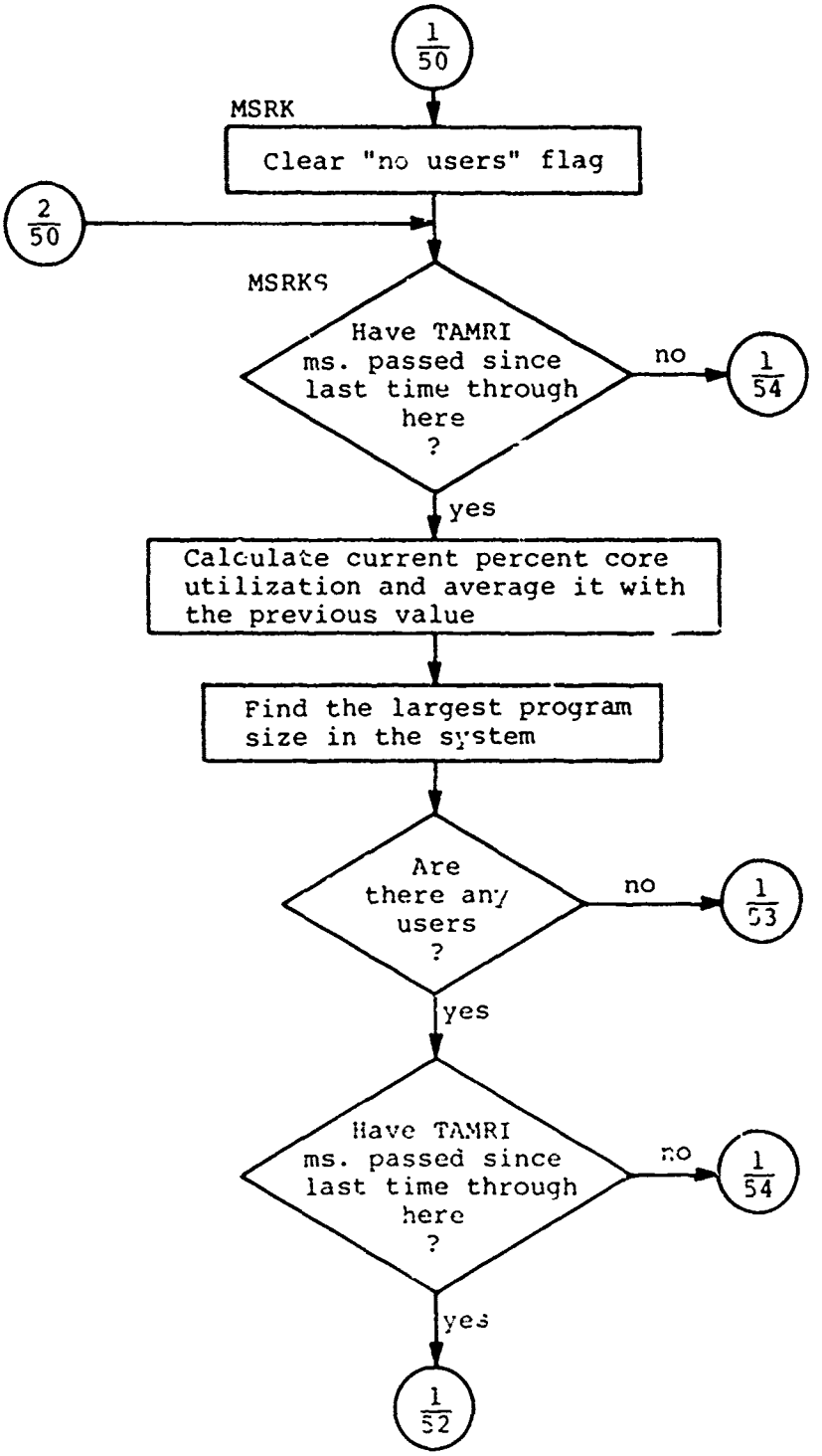

KIOCC
Set LTIN to current time
Clear "data in transmission" flag
"I/O roadblock" flag set ?
no
1/47
yes
Clear "I/O roadblocked" flag
Program in core ?
yes
Set allocator flag
no
Set MAP flag
Move UST from MEMQ to top of CPUQ
Call ATCHG to change state to "awaiting memory"
Call ATCHG to change state to "useful core residency"
1/47

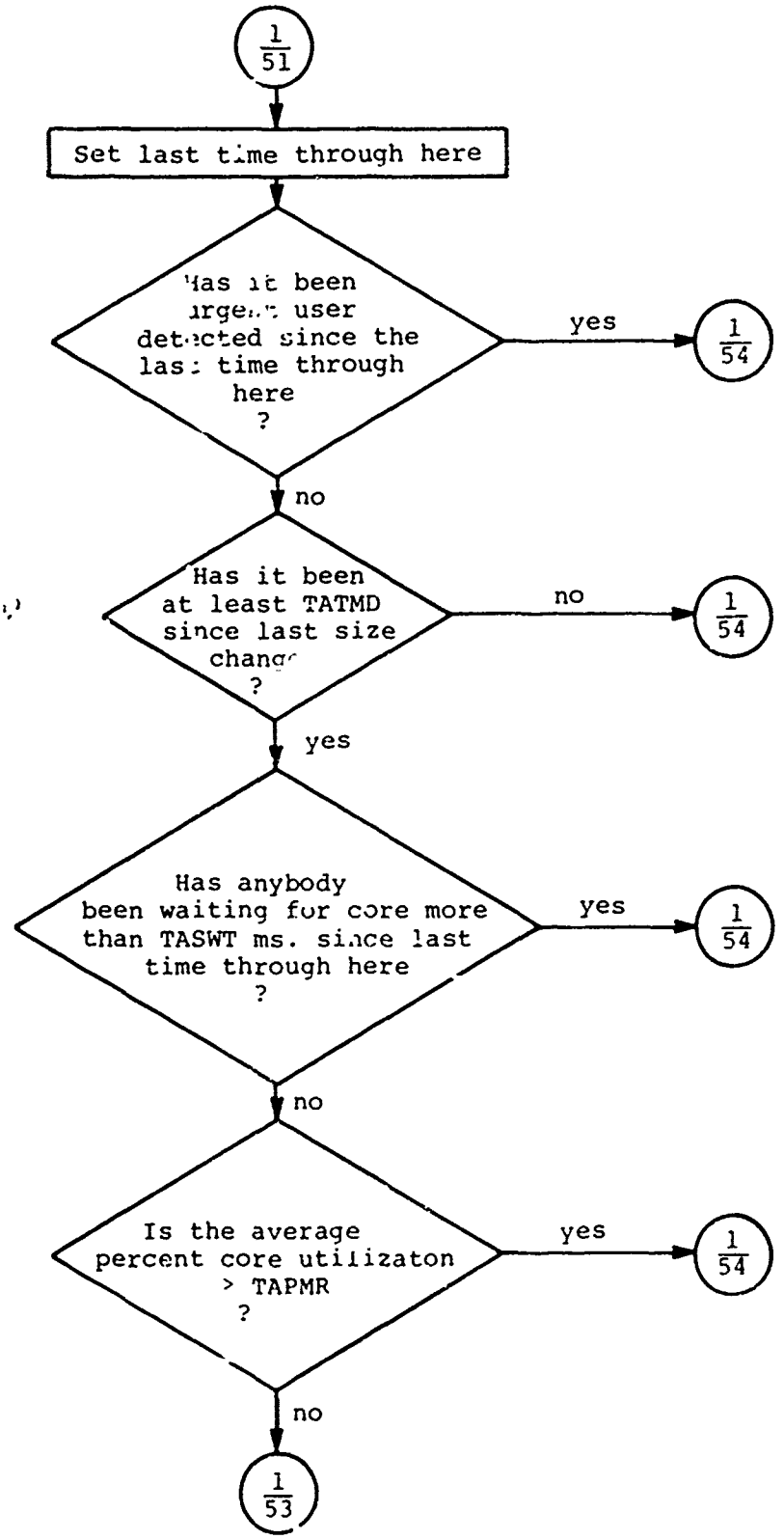

1/46
KIOCC.1
Remove this KIOCC from the CCQ and destroy it
RETURN
DIOCC
Remove the first UST from the DIOQ
Clear "disk I/O in progress" flag
Set allocator flag
Move the UST to the top of the CPUQ
Remove this DIOCC from the CCQ
DIOQ empty ?
yes
no
Destroy this DIOCC
RETURN
Schedule a courtesy call for the first UST in the DIOQ and file it in the CCQ
RETURN

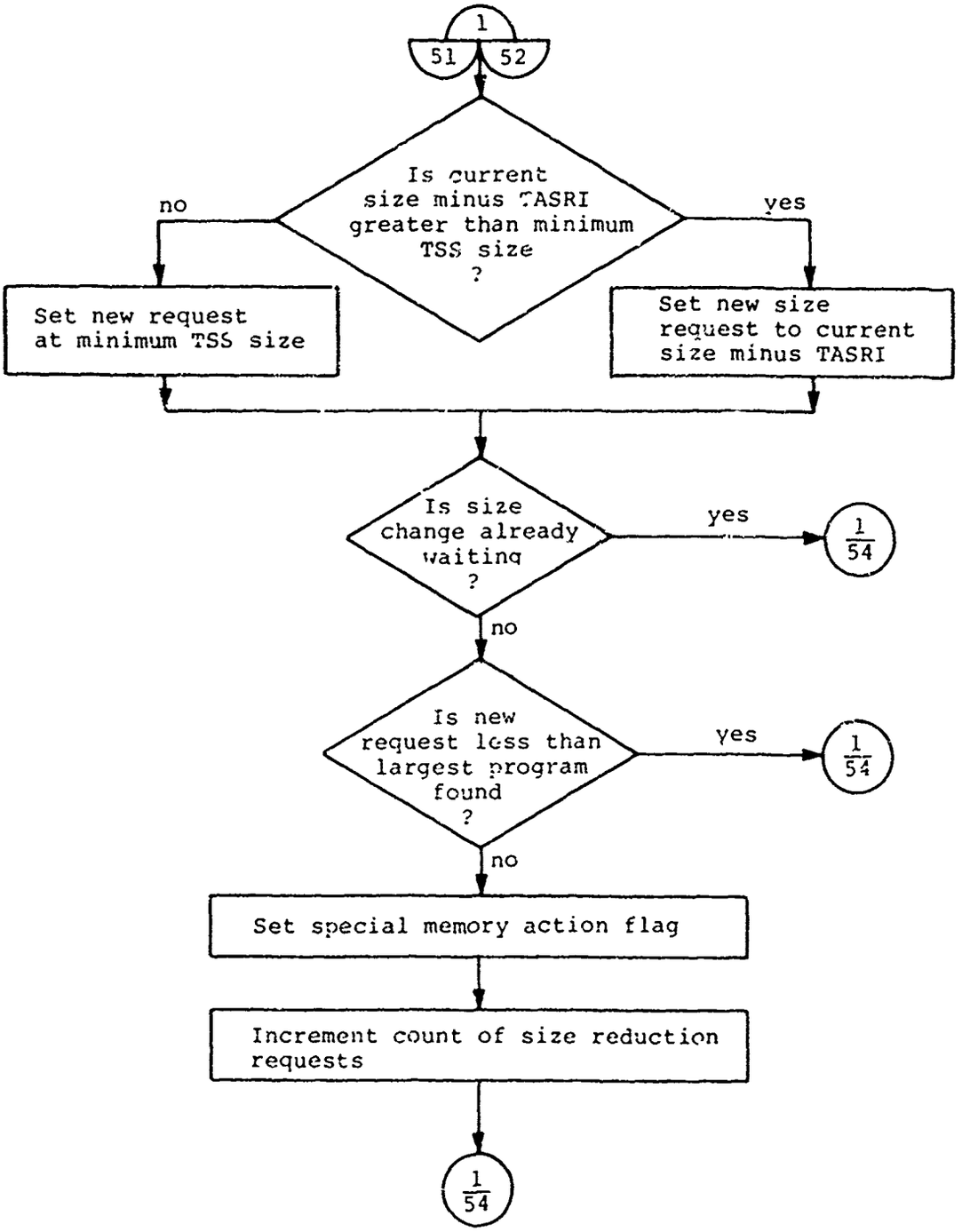

1ALLCC
Decrement number of jobs in core
Clear "swapping out" flag and set MAP flag
Data in transmission ?
yes
no
Increment count of key I/O swaps
Call ATCHG to change state to "out of core"
Was program force swapped ?
yes
no
Call ATCHG to change state to "awaiting memory"
Call ATCHG to change state to "awaiting memory after force swap"
ALLCCA
Call MBD to deallocate core
Decrement number of jobs swapping
Remove this 1ALLCC from the CCQ and destroy it
RETURN

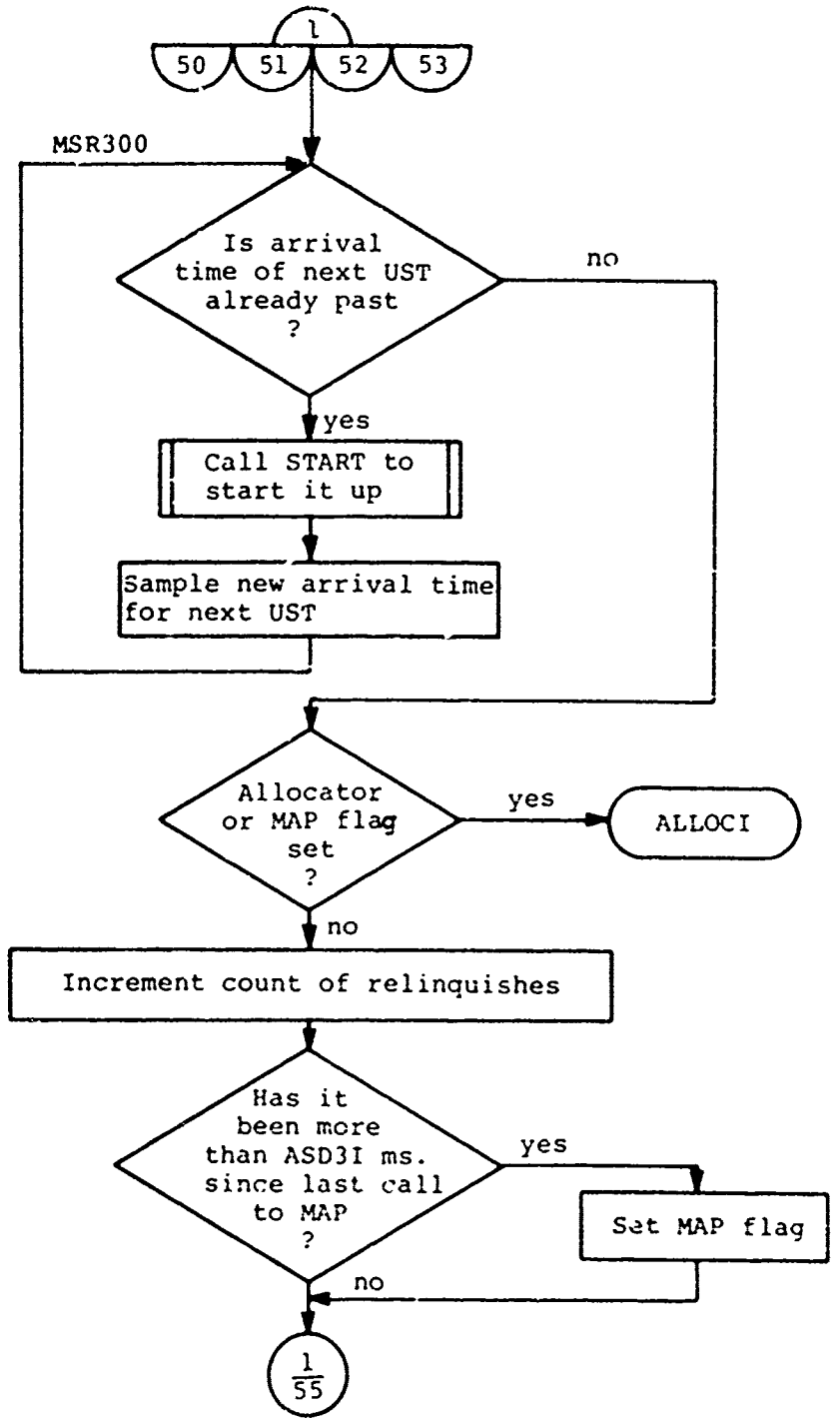

2ALLCC
Set "in core" flag and clear "swapping in" and "force swapped" flags
Set allocator flag
Move the UST from the MEMQ to the top of the CPUQ
Call ATCHG to change state to "useful core residency"
Decrement number of jobs swapping
Remove this 2ALLCC from the CCQ and destroy it
RETURN

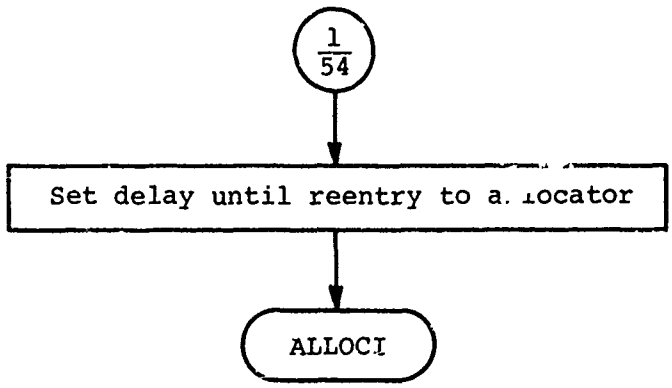

LINSV
Clear line service flag and "no users" flag
Set time of last entry into LINSV
Has it been ≥ TLNLM ms. since last time through next part ?
no
1/51
yes
Mark last time through here
Are both CPUQ and MEMQ empty ?
no
1/51
Set "no users" flag
Is TSS size at minimum ?
yes
1/54
no
2/51


1/50
MSRK
Clear "no users" flag
2/50
MSRKS
Have TAMRI ms. passed since last time through here ?
no
1/54
yes
Calculate current percent core utilization and average it with the previous value
Find the largest program size in the system
Are there any users ?
no
1/53
yes
Have TAMRI ms. passed since last time through here ?
no
1/54
yes
1/52


1/51
Set last time through here
Has it been urgent user detected since the last time through here ?
yes
1/54
no
Has it been at least TATMD since last size change ?
no
1/54
yes
Has anybody been waiting for core more than TASWT ms. since last time through here ?
yes
1/54
no
Is the average percent core utilizaton > TAPMR ?
yes
1/54
no
1/53


1
51
52
Is current size minus TASRI greater than minimum TSS size ?
no
yes
Set new request at minimum TSS size
Set new size request to current size minus TASRI
Is size change already waiting ?
yes
1/54
no
Is new request less than largest program found ?
yes
1/54
no
Set special memory action flag
Increment count of size reduction requests
1/54


1
50
51
52
53
MSR300
Is arrival time of next UST already past ?
no
yes
Call START to start it up
Sample new arrival time for next UST
Allocator or MAP flag set ?
yes
ALLOCI
no
Increment count of relinquishes
Has it been more than ASD3I ms. since last call to MAP ?
yes
Set MAP flag
no
1/55


1/54
Set delay until reentry to a. ocator
ALLOCI

Appendix 3

Program Listing

```
1    1  ''
2    2  ''          SIMULATION MODEL FOR TSS ALLOCATOR
3    3  ''                  (RELEASE 8)
4    4  ''
5    5  ''
6    6  PREAMBLE
7    7  NORMALLY MODE IS INTEGER AND DIMENSION IS 0
8    8  DEFINE EVDIAG TO MEAN FOR ES=1 TO EVSWITCH
9    9  DEFINE MS. TO MEAN UNITS
10   10 DEFINE SHOW TO MEAN PRINT 1 LINE WITH TIME.V
11   11 DEFINE RDIAG TO MEAN FOR RS=1 TO RSWITCH PRINT 1 LINE THUS
12   12 DEFINE IRIT AS A RELEASABLE ROUTINE
13   13 ''
14   14 ''      DEFINITION OF SYSTEM QUEUES AND RANDOM VARIABLES
15   15 ''
16   16 PERMANENT ENTITIES
17   17 THE SYSTEM OWNS A CPUQ, A MEMQ, A CCQ AND A DIOQ
18   18 THE SYSTEM HAS A USTIAT RANDOM LINEAR VARIABLE
19   19 THE SYSTEM HAS A NOKIN RANDOM STEP VARIABLE
20   20 THE SYSTEM HAS A NOKOUT RANDOM STEP VARIABLE
21   21 THE SYSTEM HAS A NODIO RANDOM STEP VARIABLE
22   22 THE SYSTEM HAS A SWAPOUT RANDOM LINEAR VARIABLE
23   23 THE SYSTEM HAS A CPUOUT RANDOM LINEAR VARIABLE
24   24 THE SYSTEM HAS A KIOOUT RANDOM LINEAR VARIABLE
25   25 THE SYSTEM HAS A SIZEDIST RANDOM STEP VARIABLE
26   26 DEFINE USTIAT AS A REAL, STREAM 1 VARIABLE
27   27 DEFINE NOKIN AS AN INTEGER, STREAM 2 VARIABLE
28   28 DEFINE NOKOUT AS AN INTEGER, STREAM 3 VARIABLE
29   29 DEFINE NODIO AS AN INTEGER, STREAM 4 VARIABLE
30   30 DEFINE SWAPOUT AS A REAL, STREAM 5 VARIABLE
31   31 DEFINE CPUOUT AS A REAL, STREAM 6 VARIABLE
32   32 DEFINE KIOOUT AS A REAL, STREAM 7 VARIABLE
33   33 DEFINE SIZEDIST AS AN INTEGER, STREAM 8 VARIABLE
34   34 ''
35   35 ''      DEFINITION OF UST ENTITY
36   36 ''
37   37 TEMPORARY ENTITIES
38   38 EVERY UST HAS A LTIN, A LSIZE, A LSTIO, A LSPTS, A LTMWT, A LICW,
39   39        A LTM , A LTM1, A LTM2, A LTM3, A LTM4, A LTM5, A LTMRS,
40   40        A LTC21, A LTC22, A LTC31, A LTC32, A FL18, A FL19, AN OUTCC,
41   41        A FL21, A FL22, A FL23, A FL24, A FL34, A FK19,
42   42        A NXTDIO, A NXTKIN, A NXTKOUT, A KILL, A CHKCPU,
43   43        A JOBNO, A KIOAT, A KOIAT, A DIOIAT
44   44     AND MAY BELONG TO THE CPUQ, THE MEMQ AND THE DIOQ
45   45 DEFINE LTIN, LSPTS, LTMWT, LTM0, LTM1, LTM2, LTM3, LTM4,
46   46      LTM5, LTMRS, NXTDIO, NXTKIN, NXTKOUT, KILL, NXTUST, NXTCC,
47   47      EAT, CHKCPU, KIOAT, KOIAT, DIOIAT AS REAL VARIABLES
48   48 DEFINE LSIZE, LICW, LTC21, LTC22, LTC31, LTC32, FL18, FL19, FL21,
49   49    FL22, FL23, FL24, FL34, FK19, OUTCC, JOBNO, LSTIO AS VARIABLES
50   50 DEFINE MEMO AS A SET RANKED BY LOW LSIZE
51   51 ''
```

```
 52   52  **      DEFINITION OF EVENTS
 53   53  **
 54   54  EVENT NOTICES
 55   55  INCLUDE ALLOCI, MAP, SDP, SDP3, SDP4, SDP5, SDP6, SDP7, KONDRL,
 56   56      KOTDRL, DRLDIO, DRLRET, LINSV, EXENTR, RETSSX, STERM, REINIT,
 57   57      AND PDIAG
 58   58  EVERY KIOCC HAS A KUST, BELONGS TO THE CCQ,
 59   59   HAS A S.CCQ IN WORD 7, AND HAS A P.CCQ IN WORD 8
 60   60  EVERY 1ALLCC HAS A SUST1, BELONGS TO THE CCQ,
 61   61   HAS A S.CCQ IN WORD 7, AND HAS A P.CCQ IN WORD 8
 62   62  EVERY 2ALLCC HAS A SUST2, BELONGS TO THE CCQ,
 63   63   HAS A S.CCQ IN WORD 7, AND HAS A P.CCQ IN WORD 8
 64   64  EVERY DIOCC HAS A DUST, BELONGS TO THE CCQ,
 65   65   HAS A S.CCQ IN WORD 7, AND HAS A P.CCQ IN WORD 8
 66   66  DEFINE CCQ AS A SET RANKED BY LOW TIME.A WITHOUT L.N AND M ATTRIBUTES
 67   67  **
 68   68  **      DEFINITION OF MEMORY MAP ARRAYS AND DIAGNOSTIC SWITCHES
 69   69  **
 70   70  DEFINE SJOB, SHOLE, SUC, PRED, IDPTR AS 1-DIMENSIONAL
 71   71      ARRAYS
 72   72  DEFINE HEAD, TAIL, AVAIL, EVSWITCH, CSWITCH, MSWITCH, SCSWITCH,
 73   73      RSWITCH, ES, RS, TSTRI, TKILL, TLNAA, DEVS, DRS, DQS, DMS,
 74   74      DSCS, I, J, USWITCH, KSWITCH, DUS, DKS, SSSWITCH, DSS AS VARIABLES
 75   75  **
 76   76  **      DEFINITION OF FLAGS, VARIABLES AND COUNTERS
 77   77  **
 78   78  DEFINE LSFLG, MPWF, ALOCI, TLFLG, APAP1, APAP2, AMAP,
 79   79      MPACT, 1AUPWT, 2AUPWT, AMAP2, AMN1, AMN2, ASDF, SFUSE,
 80   80      ASDPC, TAUPG, TATMN, TAAUG, TAHOL, TACOR, TSIRC, ASDP7, TSWPK,
 81   81      INCORE, ILLOCPT, WAITCOR AS VARIABLES
 82   82  DEFINE MLMALOCOK, AMBA, AMBA4, AMBA5, INDDRL, TAUSE, TEMP,
 83   83      TAPMU, TALPS, TSERC, IASIO, KOSWAP, TAGTC, PERUSED AS VARIABLES
 84   84  DEFINE MAFTM, ALUTM, ASD3T, TALUT, TTFAC, URGT, TALCT,
 85   85      T, SPARE, RESPT, TLLST, TLOLD, TASRT, TASWT, DISPT, TAGPT,
 86   86      TAGTU, TLTLM, MXTINT, DIOMEAN, INTMEAN, STOPTIME, NEXT,
 87   87      WAKET, OUTMEAN, DBEGIN, DDUR, MNDELAY, MXDELAY, INITIME AS REAL VARIABLES
 88   88  DEFINE AMFTM, TASWT, IASID, AMTC, TAGMI, TATMC, TATMD,
 89   89      TAMAW, TLALM, TAMRI, IASCF, ASD3I, TCDEL AS REAL VARIABLES
 90   90  DEFINE LASF, TAMIS, TALPF, TAMII, TAMMS, IASMS, TAPMN, TASRI,
 91   91      ITALUT,INITCORE AS VARIABLES
 92   92  **
 93   93  **      DEFINITION OF VARIABLES FOR COLLECTING STATISTICS
 94   94  **
 95   95  DEFINE KEYOUT, KEYIN, DISKIO, TSWAP, SIZEINCR, SIZERED,
 96   96      PROGSIZE, HOLSIZE, CORSIZE, USEDSIZE AS VARIABLES
 97   97  DEFINE VKIIAT, VKOIAT, VUSTIAT, VDIOIAT, VSIZE, VCFUDUE AS REAL
 98   98      VARIABLES
 99   99  DEFINE LTM0SS, LTM1SS, LTM2SS, LTM3SS, LTM4SS, LTM5SS, SSRESP,
100  100      SSCPU AS REAL VARIABLES
101  101  DEFINE SSMIN, SSKOUT, SSDIO, SSFSWAP, SSSWAPO AS VARIABLES
102  102  TALLY MPROGSIZE AS THE MEAN AND SPROGSIZE AS THE STD OF PROGSIZE
103  103  TALLY MHOLSIZE AS THE MEAN AND SHOLSIZE AS THE STD OF HOLSIZE
```

```
104  104  TALLY MCORSIZE AS THE MEAN AND SCORSIZE AS THE STD OF CORSIZE
105  105  TALLY MUSEDSIZE AS THE MEAN AND SUSEDSIZE AS THE STD OF USEDSIZE
106  106  TALLY MPERUSED AS THE MEAN AND SPERUSED AS THE STD OF PERUSED
107  107  TALLY MVKIIAT AS THE MEAN AND SVKIIAT AS THE STD OF VKIIAT
108  108  TALLY MVKOIAT AS THE MEAN AND SVKOIAT AS THE STD OF VKOIAT
109  109  TALLY MVUSTIAT AS THE MEAN AND SVUSTIAT AS THE STD OF VUSTIAT
110  110  TALLY MVDIOIAT AS THE MEAN AND SVDIOIAT AS THE STD OF VDIOIAT
111  111  TALLY MVSIZE AS THE MEAN AND SVSIZE AS THE STD OF VSIZE
112  112  TALLY MVCPUDUR AS THE MEAN AND SVCPUDUR AS THE STD OF VCPUDUR
113  113  TALLY MLTM0SS AS THE MEAN AND SLTM0SS AS THE STD OF LTM0SS
114  114  TALLY MLTM1SS AS THE MEAN AND SLTM1SS AS THE STD OF LTM1SS
115  115  TALLY MLTM2SS AS THE MEAN AND SLTM2SS AS THE STD OF LTM2SS
116  116  TALLY MLTM3SS AS THE MEAN AND SLTM3SS AS THE STD OF LTM3SS
117  117  TALLY MLTM4SS AS THE MEAN AND SLTM4SS AS THE STD OF LTM4SS
118  118  TALLY MLTM5SS AS THE MEAN AND SLTM5SS AS THE STD OF LTM5SS
119  119  TALLY MSSRESP AS THE MEAN AND SSSRESP AS THE STD OF SSRESP
120  120  TALLY MSSCPU AS THE MEAN AND SSSCPU AS THE STD OF SSCPU
121  121  TALLY MRESPT AS THE MEAN AND SRESPT AS THE STD OF RESPT
122  122  TALLY MSSKIN AS THE MEAN AND SSSKIN AS THE STD OF SSKIN
123  123  TALLY MSSKOUT AS THE MEAN AND SSSKOUT AS THE STD OF SSKOUT
124  124  TALLY MSSDIO AS THE MEAN AND SSSDIO AS THE STD OF SSDIO
125  125  TALLY MSSFSWAP AS THE MEAN AND SSSFSWAP AS THE STD OF SSFSWAP
126  126  TALLY MSSSWAPO AS THE MEAN AND SSSSWAPO AS THE STD OF SSSWAPO
127  127  ACCUMULATE MCPUQ AS THE MEAN AND SCPUQ AS THE STD OF N.CPUQ
128  128  ACCUMULATE MDIOQ AS THE MEAN AND SDIOQ AS THE STD OF N.DIOQ
129  129  ACCUMULATE MMEMQ AS THE MEAN AND SMEMQ AS THE STD OF N.MEMQ
130  130  ACCUMULATE MSFUSE AS THE MEAN AND SSFUSE AS THE STD OF SFUSE
131  131  ACCUMULATE MINCORE AS THE MEAN AND SINCORE AS THE STD OF INCORE
132  132  ACCUMULATE MTAURG AS THE MEAN AND STAURG AS THE STD OF TAURG
133  133  ACCUMULATE MWAIT AS THE MEAN AND SWAIT AS THE STD OF WAITCOR
134  134  ACCUMULATE MELIG AS THE MEAN AND SELIG AS THE STD OF ELIGCPU
135  135  END
```

```
7950T 01  08-28-75   17.504   CACI SIMSCRIPT II.5 FOR HIS 600/6000        USAF RELEASE 9.

   136     1  ''
   137     2  ''      ROUTINE TO INITIALIZE SIMULATION
   138     3  ''
   139     4  ROUTINE INIT
   140     5  SHOW THUS
BEGINNING OF SIMULATION AT  ******.***
   142     6  ''
   143     7  ''      READ INPUT PARAMETERS, CONSTANTS AND DISTRIBUTIONS
   144     8  ''
   145     9  READ STOPTIME, DEVS, DRS, DQS, DMS, DSCS, DUS, DKS, DSS, DBEGIN,
   146    10       DRUN, INITIME
   147    11  LIST STOPTIME, DEVS, DRS, DQS, DMS, DSCS, DUS, DKS, DSS, DBEGIN,
   148    12       DRUN, INITIME
   149    13  PRINT 1 LINE THUS
RANDOM NUMBER SEEDS
   151    14  FOR J=1 TO 10 DO
   152    15       READ SEED.V(J) PRINT 1 LINE WITH SEED.V(J) THUS
     **********
   154    16  LOOP
   155    17  READ AMPTM, ANSP, TASWT, TAMIS, TALPP, TASWP, TASID, TLTLM,
   156    18       TAMII, TAMMS, AMTQ, TAGMI, TATMC, TATMD, TAMAW, TLNLM,
   157    19       TASMS, TAMRI, TAPMR, TASRI, TASCP, ASD3I, TCDEL,
   158    20       INITCORE, INTMEAN, MNDELAY, MXDELAY, DIOMEAN, OUTMEAN
   159    21  LIST AMPTM, ANSP, TASWT, TAMIS, TALPP, TASWP, TASID, TLTLM,
   160    22       TAMII, TAMMS, AMTQ, TAGMI, TATMC, TATMD, TAMAW, TLNLM,
   161    23       TASMS, TAMRI, TAPMR, TASRI, TASCP, ASD3I, TCDEL,
   162    24       INITCORE, INTMEAN, MNDELAY, MXDELAY, DIOMEAN, OUTMEAN
   163    25  READ USTIAT, NOKIN, NOKOUT, NODIO, SWAPDUR, CPUDUR, KIODUR, SIZEDIST,
   164    26       CHECK
   165    27  IF CHECK NE 9999 PRINT 1 LINE THUS
*** ERROR - INPUT FORMAT INCORRECT
   167    28  STOP
   168    29  ELSE
   169    30  ''
   170    31  ''      RESERVE ARRAYS FOR MEMORY MAP
   171    32  ''
   172    33  RESERVE SJOB, SHOLE, SUC, PRED, IDPTR AS 50
   173    34  ''
   174    35  ''      INITIALIZE MEMORY MAP
   175    36  ''
   176    37  LET HEAD=1 LET TAIL=1
   177    38  LET SHOLE(1)=INITCORE LET SJOB(1)=0 LET TACOR=INITCORE
   178    39  LET AVAIL=2 LET SUC(1)=0 LET PRED(1)=0
   179    40  FOR I=2 TO 5  DO
   180    41       LET SUC(I)=I+1
   181    42  LOOP
   182    43  LET SUC(50)=0
   183    44  LET TAPMU=10. LET CORSIZE=INITCORE
   184    45  ''
   185    46  ''      TAKE FIRST SAMPLE OF UST ARRIVAL AND INTERRUPT
   186    47  ''
   187    48  LET NXTUST=USTIAT
```

```
7950T 01  08-28-75    17.504   CACI SIMSCRIPT II.5 FOR HIS 600/6000        USAF RELEASE 9.

  188    49  LET VUSTIAT=NXTUST
  189    50  LET NXTINT=EXPONENTIAL.F(INTMEAN,9)
  190    51  ''
  191    52  ''     SCHEDULE ENTRY TO ALLOCATOR AND TERMINATION EVENTS
  192    53  SCHEDULE AN ALLOCI NOW LET LSFLG=1
  193    54  SCHEDULE A STERM AT STOPTIME
  194    55  SCHEDULE A PDIAG AT DBEGIN  ''     SCHEDULE EVENT TO PRINT DIAGNOSTICS
  195    56  SCHEDULE A REINIT AT INITIME  ''  SCHEDULE EVENT TO RE-INITIALIZE COUNTERS
  196    57  RETURN END
```

```
7950T 01  08-28-75   17.504   CACI SIMSCRIPT II.5 FOR HIS 600/6000      USAF RELEASE 9.

  197     1  ''
  198     2  ''     MAIN ROUTINE
  199     3  ''
  200     4  MAIN
  201     5  PERFORM INIT RELEASE INIT
  202     6  START SIMULATION
  203     7  STOP
  204     8  END
```

```
7950T 01  08-28-75   17.504   CACI SIMSCRIPT II.5 FOR HIS 600/6000      USAF RELEASE 9.

  205     1  ''
  206     2  ''     EVENT TO START AND STOP PRINTING OF DIAGNOSTICS
  207     3  ''
  208     4  EVENT PDIAG SAVING THE EVENT NOTICE
  209     5  IF TIME.V LT BEGIN+0.5
  210     6  ''
  211     7  ''     SET SWITCHES AT START OF DIAGNOSTIC PERIOD.
  212     8       LET EVSWITCH=DEVS LET RSWITCH=DRS LET QSWITCH=DQS
  213     9       LET MSWITCH=DMS LET SCSWITCH=DSCS LET USWITCH=DUS LET KSWITCH=DKS
  214    10       LET SSSWITCH=DFS
  215    11       RESCHEDULE THIS PDIAG AT TIME.V+DDUR RETURN
  216    12  ''
  217    13  ''     RESET SWITCHES AT END OF DIAGNOSTIC PERIOD.
  218    14  ELSE LET EVSWITCH=0 LET RSWITCH=0 LET QSWITCH=0 LET MSWITCH=0
  219    15  LET USWITCH=0 LET KSWITCH=0 LET SSSWITCH=0
  220    16  LET SCSWITCH=0  DESTROY THIS PDIAG
  221    17  RETURN END
```

```
7950T 01  08-28-75   19.504   CACI SIMSCRIPT II.5 FOR HIS 600/6000      USAF RELEASE 9.

  222     1  ''
  223     2  ''     EVENT TO RE-INITIALIZE COUNTERS FOR COLLECTION OF STATISTICS
  224     3  ''
  225     4  EVENT REINIT
  226     5    CALL R.PROGSIZE CALL R.HOLSIZE CALL R.CORSIZE CALL R.USEDSIZE
  227     6    CALL R.PERUSED CALL R.VKIIAT CALL R.VKOIAT CALL R.VUSTIAT
  228     7    CALL R.VDIOIAT CALL R.VSIZE CALL R.VCPUDUR CALL R.LTM0SS
  229     8    CALL R.LTM1SS CALL R.LTM2SS CALL R.LTM3SS CALL R.LTM4SS
  230     9    CALL R.LTM5SS CALL R.SSRESP CALL R.SSCPU CALL R.RESPT
  231    10    CALL R.SSNI CALL R.SSKOUT CALL R.SSDIO CALL R.SSFSWAP
  232    11    CALL R.SSSWAPO CALL R.N.CPUQ CALL R.N.MEMQ CALL R.N.DIOQ
  233    12    CALL R.SFUSE CALL R.INCORE CALL R.TAURG CALL R.WAITCOP CALL R.ELIGCPU
  234    13  ''
  235    14  LET KEYOUT=0 LET KEYIN=0 LET DISKIO=0 LET TSWAP=0 LET TASIO=0
  236    15  LET ASOF7=0 LET SIZEINCR=0 LET SIZEFRED=0 LET TAGTU=0
  237    16  LET TSTRT=0 LET CKILL=0 LET APAP2=0 LET TAAUG=0 LET ALOCI=0
  238    17  LET AMAP=0 LET APAP1=0 LET TAGTC=0 LET TSWPK=0
  239    18  RETURN END
```

```
7950T 01  08-28-75   17.504   CACI SIMSCRIPT II.5 FOR HIS 600/6000      USAF RELEASE 9.

  240     1  ''
  241     2  ''     THIS ROUTINE TAKES A SNAPSHOT OF THE MEMORY MAP
  242     3  ''     TO OBTAIN VALUES FOR PROGRAM SIZE, HOLE SIZE,
  243     4  ''     USED SIZE, AND PERCENTAGE CORE USED.
  244     5  ''
  245     6  ROUTINE CORSAMP
  246     7  LET TOTCOR=  LET TAUSE=0
  247     8  LET I=HEAD GO TO SKIP  '' DON'T LOOK AT DUMMY PROG SIZE
  248     9  'NEXT' LET PROGSIZE=SJOB(I)
  249    10  'SKIP' LET HOLSIZE=SHOLE(I)
  250    11  LET TOTCOR=TOTCOR+SJOB(I)+SHOLE(I)
  251    12  LET TAUSE=TAUSE+SJOB(I)
  252    13  LET I=SUC(I)
  253    14  IF I NE 0 GO TO NEXT
  254    15  ELSE LET USEDSIZE=TAUSE LET PERUSED=(USEDSIZE*100)/TOTCOR
  255    16  IF SWITCH NE 0 SHOW, TOTCOR, USEDSIZE, PERUSED, MPERUSED THUS
AT ********.* TOTCOR=*** USED=***  = ***.*% (MEAN=***.*%)
  257    17  ELSE RETURN END
```

```
7950T 01  08-28-75   17.504   CACI SIMSCRIPT II.5 FOR HIS 600/6000      USAF RELEASE 9.

   258    1  ''
   259    2  ''     PRINT MEMO FOR DIAGNOSTIC AID
   260    3  ''
   261    4  ROUTINE MOPRINT
   262    5  PRINT 2 LINES THUS
UST          FLAGS          STATE      SIZE
          MEMORY QUEUE
   265    6  FOR EACH I OF MEMO PRINT 1 LINE WITH JOBNO(I), FL18(I), FL19(I),
   266    7       FL21(I), FL22(I), FL23(I), FL24(I), FL34(I),
   267    8       LTCW(I), LSIZE(I) THUS
****    * * * * * * *     ***       *
   269    9  RETURN END
```

```
79507 01  08-28-75   17.504   CACI SIMSCRIPT II.5 FOR HIS 600/6000        USAF RELEASE 9.

  270     1  ''
  271     2  ''     PRINT CPUQ FOR DIAGNOSTIC AID
  272     3  ''
  273     4  ROUTINE CQPRINT
  274     5  PRINT 2 LINE THUS
UST          FLAGS          STATE      SIZE
      PROCESSOR QUEUE
  277     6  FOR EACH I OF CPUQ PRINT 1 LINE WITH JOBNO(I), FL18(I), FL19(I),
  278     7       FL21(I), FL22(I), FL23(I), FL24(I), FL34(I),
  279     8       ITCW(I), LSIZE(I) THUS
****    * * * * * * *     ***       *
  281     9  RETURN END
```

```
7950T 01  08-28-75   17.504   CACI SIMSCRIPT II.. FOR HIS 600/6000      USAF RELEASE 9.

  282     1  ''
  283     2  ''      EVENT TO END SIMULATION
  284     3  ''
  285     4  EVENT STERM
  286     5  CALL OUTPUT  RETURN END
```

```
7950T 01  08-28-75   17.50W   CACI SIMSCRIPT II.5 FOR HIS 600/6000        USAF RELEASE 9.

  287     1  ''
  288     2  ''      ROUTINE TO PRINT SIMULATION RESULTS.
  289     3  ''
  290     4  ROUTINE OUTPUT
  291     5  SHOW THUS
SIMULATION TERMINATION AT **********.***
  293     6  ''
  294     7  ''      PRINT CORE STATISTICS
  295     8  ''
  296     9  SKIP 2 OUTPUT LINES
  297    10  PRINT 6 LINES WITH MPROGSIZE, SPROGSIZE, MHOLSIZE, SHOLSIZE,
  298    11   MCORSIZE, SCORSIZE, MUSEDSIZE, SUSEDSIZE, MPERUSED, SPERUSED THUS
CORE STATISTICS                  MEAN              STANDARD DEVIATION
PROGRAM SIZES                    ****.***          *****.*****
HOLE SIZES                       ****.***          *****.*****
SWAP AREA SIZE                   ****.***          *****.*****
TOTAL USED CORE                  ****.***          *****.*****
PERCENT CORE USED                ****.***          *****.*****
  305    12  ''
  306    13  ''      PRINT MEAN RATES AND COUNTS
  307    14  ''
  308    15  LET T=(TIME.V-INITIME)/3600000.0  SKIP 2 OUTPUT LINES
  309    16  PRINT 2  LINES WITH KEYOUT/I, KEYIN/I, DISKIO/I, TSWAP/I, TASIO/I,
  310    17   KOSWAP/I, AFOPT/I, SIZEINCR/I, SIZEDEC/I, TAGTU/I, TSTRT/I, TKILL/I,
  311    18   APAP2/I, MAPUS/I, ALOCI/I, AMAP/I, APAP1/I, TAGTC/I, TSWPK/I THUS
OVERALL MEAN RATES (PER HOUR)
MAPPED OUTPUTS                   ******.**
MAPPED INPUTS                    ******.**
DISK I/OS                        ******.**
SWAP OUTS                        ******.**
SWAP OUTS (KEY I/O)              ******.**
SWAP OUTS (KEY OUTPUT)           ******.**
FORCE SWAPS                      ******.**
SIZE INCREASES                   ******.**
SIZE DECREASES                   ******.**
SUBSYSTEM CPU TIME               **********.*** MS.
SUBSYSTEM STARTS                 ******.**
SUBSYSTEM KILLS                  ******.**
SUBSYSTEM DISPATCHES             ******.**
RECENT USERS                     ******.**
ENTRIES TO PAP                   ******.**
ENTRIES TO MAP                   ******.**
ALLOCATOR IDLES                  ******.**
INTERRUPTS                       ******.**
TOTAL CORE SWAPPED               ******.** K
  332    19  ''
  333    20  ''      PRINT SUBSYSTEM STATISTICS
  334    21  ''
  335    22  SKIP 2 OUTPUT LINES
  336    23  PRINT 22 LINES WITH MVUSTIAT, SVUSTIAT, MVKIIAT, SVKIIAT, MVKOIAT,
  337    24   SVKOIAT, MVDIOIAT, SVDIOIAT, MVSIZE, SVSIZE, MVCPUDUR, SVCPUDUR,
  338    25   MSSCPU, SSSCPU, MSSRESP, SSSRESP, MRESPT, SRESPT,
```

```
795CT C1  08-28-75   17.504   CACI SIMSCRIPT II.5 FOR HIS 600/6000        USAF RELEASE 9.

  339   26   MLTM0SS, SLTM0SS, MLTM1SS, SLTM1SS, MLTM2SS,
  340   27   SLTM2SS, MLTM3SS, SLTM3SS, MLTM4SS, SLTM4SS, MLTM5SS, SLTM5SS.
  341   28   MSSKIN, SSSKIN, MSSKOUT, SSSKOUT, MSSDIO, SSSDIO, MSSFSWAP,
  342   29   SSSFSWAP, MSSSWAPO, SSSSWAPO THUS
SUBSYSTEM STATISTICS              MEAN          STANDARD DEVIATION
SUBSYSTEM IAT                  ******.**        ******.**
KEY INPUT IAT                  ******.**        ******.**
KEY OUTPUT IAT                 ******.**        ******.**
DISK I/O IAT                   ******.**        ******.**
PROGRAM SIZE                   ******.**        ******.**
CPU TIME (SAMPLED)             ******.**        ******.**
CPU TIME (USED)                ******.**        ******.**
RESPONSE TIME (R19)            ******.**        ******.**
RESPONSE TIME (INDIVIDUAL)     ******.**        ******.**
TIME IN STATE
   NON-USEFUL CORE          ********.**      **********.**
   SWAP                     ********.**      **********.**
   USEFUL CORE              ********.**      **********.**
   OUT OF CORE              ********.**      **********.**
   WAIT MEMORY              ********.**      **********.**
   WAIT MEMORY AFTER FS     ********.**      **********.**
NO. OF KEY INS                 ******.**        *********.**
NO. OF KEY OUTS                ******.**        *********.**
NO. OF DISK I/O'S              ******.**        *********.**
NO. OF FORCE SWAPS             ******.**        *********.**
NO. OF SWAPS                   ******.**        *********.**
  365   30   ''
  366   31   ''      PRINT QUEUE STATISTICS
  367   32   ''
  368   33   SKIP 2 OUTPUT LINES
  369   34   PRINT 9 LINES WITH NCPUQ, SCPUQ, MMEMQ, SMEMQ, MDIOQ, SDIOQ,
  370   35        MSFUSE, SSFUSE, MINCORE, SINCORE, MTAURG, STAURG, MWAIT, SWAIT,
  371   36        MELIG, SELIG THUS
QUEUE LENGTHS                     MEAN          STANDARD DEVIATION
PROCESSOR QUEUE                ****.**          *****.**
MEMORY QUEUE                   ****.**          *****.**
DISK I/O QUEUE                 ****.**          *****.**
USERS SWAPPING                 ****.**          *****.**
USERS IN CORE                  ****.**          *****.**
ELIGIBLE USERS                 ****.**          *****.**
USERS WAITING FOR CORE         ****.**          *****.**
USERS ELIGIBLE FOR CPU         ****.**          *****.**
  381   37   STOP END
```

```
7950I 01  08-28-75   17.504   CACI SIMSCRIPT II.5 FOR HIS 600/6000        USAF RELEASE 9.

  382     1  ''
  383     2  ''     THIS ROUTINE IS CALLED IF THE SIMULATION IS TERMINATED
  384     3  ''     ABNORMALLY FOR ANY REASON. IT PRINTS CERTAIN VALUABLE
  385     4  ''     DATA FOR DEBUGGING AND LOCATING A POSSIBLE ERROR.
  386     5  ''
  387     6  ROUTINE SNAP.R
  388     7  ''
  389     8  ''     PRINT EVENT TIMES FOR EVERY EVENT
  390     9  FOR I=1 TO EVENTS.V DO
  391    10       PRINT 1 LINE WITH I, TIME.A(F.EV.S(I)) THUS
 ****  *********.********
  393    11       LOOP
  394    12  LIST TIME.V, NXTUST, NXTDIO, NXTKIN, NXTKOUT, KILL, NXTINT
  395    13  ''
  396    14  ''     PRINT ALL WAITING COURTESY CALL TIMES
  397    15  FOR EACH I OF CC0 PRINT 1 LINE WITH TIME.A(I) THUS
 ********.*******
  399    16  ''
  400    17  ''     PRINT SIMULATION OUTPUT UPTO THIS POINT.
  401    18  CALL OUTPUT
  402    19  STOP END
```

```
7950T 01  08-28-75   17.504   CACI SIMSCRIPT II.5 FOR HIS 600/6000      USAF RELEASE 9.

   403     1  ''
   404     2  ''      ENTRY TO ALLOCATOR
   405     3  ''
   406     4  EVENT ALLOCI
   407     5  EVDIAG SHOW THUS
ALLOCI AT ********.*
   409     6  IF LSFLG EQ 1 SCHEDULE A LINSV IN .006 MS. RETURN
   410     7  ELSE IF HPWF EQ 1 SCHEDULE A MAP IN .01 MS. RETURN
   411     8  ''
   412     9  ''      SCHEDULE NEXT SUBSYSTEM IN CPUQ
   413    10  ELSE 'PAP' ADD 1 TO ALOCI
   414    11  LET TLFLG=1 LET EXT=.04
   415    12  'PAP.1' FOR EACH UST OF CPUQ DO
   416    13      ADD .02 TO EXT
   417    14      IF FL18+FL19+FL21+FL23 EQ 0
   418    15          ADD 1 TO APAP2
   419    16          IF FL34 EQ 1 SCHEDULE AN SDP7 IN EXT+.022 MS. RETURN
   420    17          ELSE SCHEDULE A RETSSX IN EXT+0.1 MS.
   421    18          LET FL24=0
   422    19          RETURN
   423    20      ELSE LOOP
   424    21  ''
   425    22  ''      ALLOCATOR IS IDLE. CALL LINE SERVICE
   426    23  LET TLFLG=0
   427    24  ADD 1 TO APAP1
   428    25  SCHEDULE A LINSV IN EXT MS.
   429    26  RETURN END
```

```
7950T 01   08-28-75    17.504    CACI SIMSCRIPT II.5 FOR HIS 600/6000        USAF RELEASE 9.

  430     1  **
  431     2  **      MEMORY ALLOCATOR PROCESS
  432     3  **
  433     4  EVENT MAP
  434     5  EVDIAG SHOW THUS
MAP     AT ********.*
  435     6  ADD 1 TO AMAP LET EXT=0
  437     7  LET MPNF=0
  438     8  **
  439     9  **      SEE IF SPECIAL MEM ACTION WAITING
  440    10  IF MPACT EQ 1 CALL SPMACT
  441    11  ELSE LET MAPTM=TIME.V
  442    12  **      CHECK UP ON URGENT USER IF ANY
  443    13  IF 1AURWT EQ 0 ADD .026 TO EXT GO TO MAP.3
  444    14  ELSE IF 2AURWT LE SHOLE(TAIL) ADD .039 TO EXT GO TO MAP.2A
  445    15  ELSE ADD .046 TO EXT IF TIME.V-ALUTM LE AMPTM GO TO MAP.3
  446    16  ELSE ADD 1 TO AMAP2
  447    17  LET 1AURWT=0 LET 2AURWT=0
  448    18  **
  449    19  **      FIND USERS TO SWAP OUT
  450    20  'MAP.3' LET R6=0
  451    21  FOR EACH UST OF MEMO DO
  452    22      IF FL19+FL21+FL22+FL23 EQ 0
  453    23          ADD .03 TO EXT
  454    24          IF FL34 EQ 0 GO TO MAP.4
  455    25          ELSE IF R6 EQ 0 LET R6=UST
  456    26      ELSE ELSE LOOP
  457    27  **
  458    28  **      DID WE DETECT A FORCE SWAP?
  459    29  'MAP.3B' IF R6 EQ 0 IF MPACT EQ 0 SCHEDULE AN ALLOCI IN EXT+0.01 MS.
  460    30          RETURN
  461    31      ELSE SCHEDULE AN SDP IN EXT MS.
  462    32      LET AMN2=1
  463    33  RETURN
  464    34  ELSE LET UST=R6
  465    35  **
  466    36  **      TRY TO GET CORE FOR JOB
  467    37  'MAP.4' CALL MBA
  468    38  **
  469    39  **      IF UNSUCCESSFUL, CALL SWAP DECISION PROCESSOR
  470    40  IF MEMALOCOK EQ 0 SCHEDULE AN SDP IN EXT+0.016 MS.
  471    41      ADD 1 TO ASDP
  472    42      LET AMN2=LSIZE
  473    43      RETURN
  474    44  **
  475    45  **      SUCCESSFUL, SWAP JOB IN
  476    46  ELSE CALL SWIN
  477    47  RESCHEDULE A MAP IN EXT+0.03 MS.
  478    48  RETURN
  479    49  **
  480    50  **      URGENT USER GOT CORE, TAKE DOWN CORE FENCE
  481    51  'MAP.2A' LET UST=1AURWT
```

```
7950T 01  08-28-75   17.504   CACI SIMSCRIPT II.5 FOR HIS 600/6000      USAF RELEASE 9.

 482   52  ADD .013 TO EXT
 483   53  LET 1AUPWT=0
 484   54  LET 2AUPWT=0
 485   55  IF FL19+FL21+FL22+FL23 NE 0 GO TO MAP.3
 486   56  ELSE IF UST IS NOT IN MEMQ GO TO MAP.3
 487   57  ELSE GO TO MAP.4
 488   58  END
```

```
7950T 01  08-28-75   17.504   CACI SIMSCRIPT II.5 FOR HIS 600/6000      USAF RELEASE 9.

  489     1  ''
  490     2  ''     SWAP DECISION PROCESSOR
  491     3  ''
  492     4  ''     SWAP ALL JOBS > OR = SIZE REQUIRED
  493     5  EVENT SDP
  494     6  EVDIAG SHOW THUS
SDP     AT ********.*
  496     7  IF SPUSE GE 2*LNSF IF MPACT EQ 0 SCHEDULE AN ALLOCI IN 0.04 MS.
  497     8      RETURN
  498     9      ELSE ELSE LET AMN1=0
  499    10  ''
  500    11  ''     SEARCH MEMORY QUEUE FOR KEY I/O ROADBLOCKED JOBS
  501    12  LET EXT=0.05 FOR EVERY UST OF MEMQ DO
  502    13      IF FL19+FL22 EQ 2 AND FL21+FL23 EQ 0
  503    14          ADD 0.024 TO EXT
  504    15          IF USIZE GE AMN2 GO TO SDP.2E
  505    16          ELSE LET AMN1=UST
  506    17      ELSE LOOP
  507    18  ''
  508    19  ''     DID WE FIND ANY JOB TO SWAP OUT?
  509    20  IF AMN1 EQ   SCHEDULE AN SDP3 IN EXT MS.
  510    21  RETURN
  511    22  ''
  512    23  ''     SWAP OUT JOB FOUND.
  513    24  ELSE LET UST=AMN1
  514    25  'SDP.2E' CALL SWOUT
  515    26  LET AMN1=0
  516    27  ''
  517    28  ''     REPEAT FOR MORE JOBS.
  518    29  RESCHEDULE AN SDP IN EXT MS.
  519    30  RETURN END
```

```
 520    1  **
 521    2  **    SCAN MEMORY QUEUE FOR URGENT USERS
 522    3  **
 523    4  EVENT SDP3
 524    5  EVDIAG SHOW THUS
SDP3   AT ********.*
 526    6  IF TIME.V-ASD3T LE ASD3I
 527    7      ADD 1 TO ASD3C
 528    8      IF ASD3C LT 6 SCHEDULE AN ALLOCI IN .017 MS.
 529    9        RETURN
 530   10      ELSE LET ASD3C=0
 531   11  ELSE
 532   12  LET ASD3T=TIME.V
 533   13  LET I=0
 534   14  LET TAIMN=0
 535   15  LET TALUT=0
 536   16  **
 537   17  **    SCAN ENTIRE  MEMORY QUEUE
 538   18  LET EXT=0.04
 539   19  FOR EVERY UST OF MEMQ DO
 540   20      IF FL19+FL21+FL22+FL23 NE 0 GO TO SDP.3X
 541   21      ELSE ADD 0.057 TO EXT
 542   22      IF TIME.V-LTSWT LE TASWT GO TO SDP.3X
 543   23      ELSE
 544   24  **
 545   25  **    CALCULATE JOB WAIT FACTOR DEPENDING
 546   26  **    ON SIZE AND TIME ALREADY WAITED
 547   27      'SDP.3A' LET WTFAC=LSIZE
 548   28      IF TAIMN=TATMN+LSIZE
 549   29      IF LSIZE GT TAMIS LET WTFAC=WTFAC*TALPP
 550   30      ELSE LET WTFAC=WTFAC/TASWT
 551   31      IF WTFAC LE TASWT LET WTFAC=TASWT
 552   32      ELSE IF TIME.V-LTSWT LT WTFAC GO TO SDP.3X
 553   33  **
 554   34  **    SEE IF HE WAS SCHEDULED TO BE FORCE SWAPPED.
 555   35      ELSE IF FL34 EQ 1
 556   36          LET FL34=0
 557   37          CALL ATCHG(4)
 558   38          ADD .02 TO EXT
 559   39          GO TO SDP.3X
 560   40      ELSE
 561   41  **
 562   42  **    REGISTER HIM AS AN URGENT USER.
 563   43      ADD 1 TO I
 564   44      ADD 1 TO TAURG
 565   45      LET URGT=TIME.V-LTSWT-WTFAC
 566   46      IF USWITCH NE 0 PRINT 1 LINE WITH JOBNO, URGT, TIME.V THUS
     **** FOUND URGENT FOR ********.* MS. AT ********.**
 568   47      ELSE IF URGT GE TALUT
 569   48      LET TALUT=URGT LET ITALUT=UST
 570   49      ELSE ADD 0.02 TO EXT
 571   50      'SDP.3X' LOOP
```

```
7950T 01  08-28-75   17.504   CACI SIMSCRIPT II.5 FOR HIS 600/6000      USAF RELEASE 9.

  572    51  LET TAURG=I
  573    52  SCHEDULE AN SDP4 IN EXT HS. RETURN
  574    53  END
```

```
7950T 01  08-28-75   17.504   CACI SIMSCRIPT II.5 FOR HIS 600/6000      USAF RELEASE 9.

  575    1  **
  576    2  **     DECIDE WHETHER TO GROW TSS SIZE
  577    3  **
  578    4  EVENT SDP4
  579    5  EVDIAG SHOW THUS
SDP4    AT ********.*
  581    6  IF TALUT EQ 0 SCHEDULE AN SDP6 IN 0.005 MS.
  582    7  RETURN
  583    8  ELSE IF TALUT LT TASID SCHEDULE AN SDP5 IN 0.009 MS.
  584    9  RETURN
  585   10  ELSE IF TIME.V-TALCT LT TASCF SCHEDULE AN SDP5 IN 0.017 MS.
  586   11  RETURN
  587   12  ELSE IF TAHOL GE TACOR SCHEDULE AN SDP5 IN 0.028 MS.
  588   13  RETURN
  589   14  ELSE
  590   15  **
  591   16  **     FIND NEW SIZE REQUEST
  592   17  LET TAHOL=TACOR+TAMII
  593   18  IF TAHOL GT TAMMS LET TAHOL=TACOR
  594   19       IF TAHOL GE TAMMS SCHEDULE AN SDP5 IN .048 MS.
  595   20            LET TAHOL=0 RETURN
  596   21       ELSE
  597   22       LET TAHOL=TAMMS
  598   23  ELSE
  599   24  ADD 1 TO TSIPC
  600   25  LET MPACT=1
  601   26  SCHEDULE AN SDP5 IN .057 MS.
  602   27  RETURN END
```

```
79507 01  08-28-75   17.504   CACI SIMSCRIPT II.5 FOR HIS 600/6000        USAF RELEASE 9.

  603     1  ''
  604     2  ''     SET UP CORE FENCE FOR URGENT USER
  605     3  ''
  606     4  EVENT SDP5
  607     5  EVDIAG SHOW THUS
SDP5    AT ********.*
  609     6  LET EXT=0.005
  610     7  IF 2,UNIT GT 0 GO TO SDP.5B
  611     8  ELSE LET UST=1ALUT
  612     9  IF LSIZE GT FACTOR CALL MEM ADD 0.007 TO EXT GO TO SDP.5B
  613    10  ELSE
  614    11  LET 1ALUNIT=UST
  615    12  LET 2ALUNIT=LSIZE
  616    13  LET ALUTH=TIME.V
  617    14  ADD 0.05 TO EXT
  618    15  'SDP.5B' SCHEDULE AN SDP6 IN EXT MS.
  619    16  RETURN END
```

```
7950T 01  08-28-75   17.504   CACI SIMSCRIPT II.5 FOR HIS 600/6000        USAF RELEASE 9.

  620    1  ''
  621    2  ''     CHECK JOBS TO FORCE SWAP
  622    3  ''
  623    4  EVENT SDP6
  624    5  EVDIAG SHOW THUS
SDP6   AT ********.*
  626    6  LET EXT=0.013
  627    7  IF MPACT GT . GO TO SDP.6A
  628    8  ELSE IF TALUI NE 0
  629    9       IF SPUSE LT 2*LNSF GO TO SDP.6A
  630   10       ELSE
  631   11  ELSE SCHEDULE AN ALLOCI IN 0.015 MS.
  632   12  RETURN
  633   13  'SDP.6:' FOR EVERY UST OF CPUQ DO
  634   14  ADD 0.013 TO EXT
  635   15  IF TIME.V-LTWT GT AMTQ
  636   16       IF FL24+FL34 EQ 0 LET FL34=1
  637   17            IF FL18+FL19 GT 0 SCHEDULE AN ALLOCI IN EXT+0.015 MS.
  638   18                 RETURN
  639   19            ELSE SCHEDULE AN SDP7 IN EXT+0.015 MS.
  640   20                 RETURN
  641   21       ELSE
  642   22  ELSE LOOP
  643   23  SCHEDULE AN ALLOCI IN EXT MS.
  644   24  RETURN END
```

```
79507 01  08-28-75   17.504   CACI SIMSCRIPT II.5 FOR HIS 600/6000        USAF RELEASE 9.

  645    1  ''
  646    2  ''     DUMP OUT FORCE SWAP JOBS
  647    3  ''
  648    4  EVENT SDP7
  649    5  EVDIAG SHO[illegible].JOBNO THUS
SDP7    AT ********.*  FOR ****
  651    6  LET EXT=0.11
  652    7  CALL SHOUT
  653    8  ADD 1 TO LTC31
  654    9  ADD 1 TO ASDP7
  655   10  REMOVE THIS UST FROM CPUQ
  656   11  LET TLFLG=1 LET MPWF=1 FILE THIS UST IN MZMQ
  657   12  IF CSWITCH NE 0 CALL MCFRINT CALL COFRINT ELSE
  658   13  IF CSWITCH NE 0 PRINT 1 LINE WITH JOLNO, TIME.V THUS
      *** FORCE SWAPPED AT ********.**
  660   14  ELSE SCHEDULE AN ALLOCI IN EXT MS.
  661   15  RETURN END
```

```
79507 01  08-28-75   17.504   CACI SIMSCRIPT II.5 FOR HIS 600/6000       USAF RELEASE 9.

  662     1  ''
  663     2  ''     PERFORM SPECIAL MEMORY ACTION
  664     3  ''     (TSS MEMORY INCREASE OR DECREASE)
  665     4  ''
  666     5  ROUTINE SPMACT
  667     6  RDIAG
         SPMACT CALLED
  668     7  ADD 0.011 TO EXT
  670     8  IF TAHOL EQ 0
  671     9     LET MPACT=0
  672    10     RETURN
  673    11  ELSE IF TAHOL LE TACOR GO TO SPM.3
  674    12  ''
  675    13  ''     INCREASE TSS CORE SIZE
  676    14  ELSE ADD 0.17 TO EXT IF TIME.V-TALCT LT TAGMI RETURN
  677    15  ELSE ADD 1 TO SIZEINCR ADD 0.8 TO EXT
  678    16  IF SCSWITCH NE 0 PRINT 1 LINE WITH TAHOL, TIME.V THUS
SIZE INCREASED TO ******K AT ********.*
  680    17  ELSE GO TO SPM.4
  681    18  ''
  682    19  ''     DECREASE TSS CORE SIZE
  683    20  'SPM.3' IF TAHOL EQ TACOR ADD 0.017 TO EXT GO TO SPM.4A
  684    21  ELSE IF TAHOL LE 3 GO TO SPM.4B
  685    22  ELSE IF SHOLE(TAIL) LT TACOR-TAHOL GO TO SPM.5
  686    23  ELSE LET T=TIME.V-SPMFR
  687    24  IF T GE TATMG GO TO SPM.5
  688    25  ELSE IF T LT TAIMD GO TO SPM.5
  689    26  ELSE ADD 1 TO SIZEDEC ADD 0.22 TO EXT
  690    27  IF SCSWITCH NE 0 PRINT 1 LINE WITH TAHOL, TIME.V THUS
SIZE DECREASED TO ******K AT ********.*
  692    28  ELSE
  693    29  ''
  694    30  ''     UPDATE CORE SIZE AND MEMORY MAP
  695    31  'SPM.4' LET SHOLE(TAIL)=SHOLE(TAIL)-(TACOR-TAHOL) LET TACOR=TAHOL
  696    32  'SPM.4A' LET TAHOL=0
  697    33  LET MPACT=0
  698    34  LET SPMFR=0
  699    35  'SPM.4B' LET TALCT=TIME.V
  700    36  RETURN
  701    37  ''
  702    38  ''     SEE IF REQUEST HAS WAITED TOO LONG WITHOUT ACTION
  703    39  'SPM.5' IF SPMFR EQ 0 LET SPMFR=TIME.V
  704    40  ELSE LET T=TIME.V-SPMFR ADD 0.017 TO EXT
  705    41  IF T GE TATMG GO TO SPM.4A
  707    42  ELSE GO TO SPM.4B
  707    43  END
```

```
'9507 01  08-28-75   17.504   CACI SIMSCRIPT II.5 FOR HIS 600/6000      USAF RELEASE 9.

  708     1  ''
  709     2  ''     MEMORY BUFFER ALLOCATOR
  710     3  ''
  711     4  ROUTINE MBA
  712     5  PDIAG
        MBA CALLED
  714     6  LET MEMALOCOK=0
  715     7  ADD 1 TO AMBA
  716     8  ''
  717     9  ''     TRY FINDING HOLE BIG ENOUGH IN MEMORY MAP
  718    10  LET I=HEAD
  719    11  'MBA.1' IF UST=IDPTR(I)
  720    12  PRINT 1 LINE AS FOLLOWS
### ERROR - MBA CALLED FOR UST ALREADY IN MEMORY
  722    13  STOP
  723    14  ELSE IF SHOLE(I) GE LSIZE GO TO MBA.4
  724    15  ELSE
  725    16  'MBA.2' LET I=SUC(I)
  726    17  IF I NE J GO TO MBA.1
  727    18  ''
  728    19  ''     UNSUCCESSFUL, SEE IF MEMORY CAN BE GROWN
  729    20  ELSE IF LSIZE LE TACOR ADD 0.057 TO EXT RETURN
  730    21  ELSE CALL MBA3
  731    22  RETURN
  732    23  ''
  733    24  ''     SEE IF HE WAS THE URGENT USER
  734    25  'MBA.4' ADD 1 TO AMBA4
  735    26  IF UST EQ 1AURWT
  736    27      LET 1AURWT=0
  737    28      LET 2AURWT=0
  738    29  ELSE IF I EQ TAIL
  739    30      IF SHOLE(I)-LSIZE LT 2AURWT GO TO MBA.2
  740    31      ELSE
  741    32  ELSE ADD 1 TO AMBA5
  742    33  ''
  743    34  ''     BUILD NEW ENTRY IN MEMORY MAP
  744    35  IF AVAIL EQ   PRINT 1 LINE AS FOLLOWS
### ERROR - NO MORE AVAILABLE BLOCKS FOR MEMORY MAP ARRAY
  746    36  STOP
  747    37  ELSE LET PRED(AVAIL)=I
  748    38  IF I NE TAIL
  749    39  LET PRED(SUC(I))=AVAIL
  750    40  ELSE LET TEMP=AVAIL
  751    41  LET AVAIL=SUC(AVAIL)
  752    42  LET SUC(TEMP)=SUC(I)
  753    43  LET SUC(I)=TEMP
  754    44  LET IDPTR(TEMP)=UST
  755    45  LET SIZE(TEMP)=LSIZE
  756    46  LET SHOLE(TEMP)=SHOLE(I)-LSIZE
  757    47  LET SHOLE(I)=0
  758    48  IF I EQ TAIL
  759    49      LET TAIL=TEMP
```

```
7950T 01  08-28-75   17.504   CACI SIMSCRIPT II.5 FOR HIS 600/6000      USAF RELEASE 9.

   760    50  ELSE LET MEMALOCOK=1
   761    51  CALL MMV
   762    52  ADD 0.13 TO EXT
   763    53  RETURN END
```

```
7950T 01  08-28-75   17.504   CACI SIMSCRIPT II.5 FOR HIS 600/6000        USAF RELEASE 9.

 764     1  ''
 765     2  ''     SEE IF WE CAN INCREASE TSS CORE SIZE
 766     3  ''
 767     4  ROUTINE MBA3
 768     5  RDIAG
        MBA3 CALLED
 770     6  IF TIME.V-LTWT GE TAMAW
 771     7       ADD 0.025 TO EXT
 772     8       RETURN
 773     9  ELSE IF TAHOL GE LSIZE
 774    10       ADD 0.037 TO EXT
 775    11       RETURN
 776    12  ELSE LET TAHOL=LSIZE
 777    13  LET MPACT=1
 778    14  IF TAHOL-TACOR GE TAMII
 779    15       ADD 0.057 TO EXT
 780    16       RETURN
 781    17  ELSE IF TACOR+TAMII GT TAMMS LET TAHOL=TAMMS
 782    18  ELSE ADD 0. 8 TO EXT
 783    19  RETURN END
```

```
7950T 0:  08-28-75   17.504   CACI SIMSCRIPT II.5 FOR HIS 600/6000        USAF RELEASE 9.

  784     1  ''
  785     2  ''     MEMORY BUFFER DEALLOCATOR
  786     3  ''
  787     4  ROUTINE MBD
  788     5  RDIAG
        MBD CALLED
  790     6  ''
  791     7  ''     SEARCH MEMORY MAP FOR UST TO BE DEALLOCATED
  792     8  ADD 0.033 TO EXT
  793     9  LET I=HEAD
  794    10  'MBD.1A' IF IDPTR(I) NE UST LET I=SUC(I)
  795    11       IF I EQ 0
  796    12            RETURN
  797    13       ELSE GO TO MBD.1A
  798    14  ''
  799    15  ''     REMOVE UST AND UPDATE MEMORY MAP
  800    16  ELSE LET SUC(PRED(I))=SUC(I)
  801    17  IF I NE TAIL
  802    18       LET PRED(SUC(I))=PRED(I)
  803    19  ELSE IF I EQ TAIL
  804    20       LET TAIL=PRED(I)
  805    21  ELSE ADD SHOLE(I)+SJOB(I) TO SHOLE(PRED(I))
  806    22  LET SUC(I)=AVAIL
  807    23  LET AVAIL=I
  808    24  IF SJOB(I) GE 2 LET MPWF=1 ELSE
  809    25  CALL PMV
  810    26  ADD 0.055 TO EXT
  811    27  RETURN END
```

```
7950T 01  08-28-75   17.504   CACI SIMSCRIPT II.5 FOR HIS 600/6000        USAF RELEASE 9.

  812     1  **
  813     2  **     VERIFY MEMORY MAP AND CALCULATE CORE STATISTICS
  814     3  **
  815     4  ROUTINE MMV
  816     5  RDIAG
       MMV CALLED
  818     6  ADD 0.019 TO EXT
  819     7  IF MSWITCH NE 0 PRINT 2 LINES THUS
             MEMORY MAP
 UST         PROGRAM      HOLE
  822     8  ELSE LET TOTCOR=0
  823     9  LET I=HEAD   **  START SCAN OF MEMORY MAP
  824    10  'NEXT' ADD 1.013 TO FXT
  825    11  IF MSWITCH NE 0 PRINT 1 LINE WITH JOBNO(IDPTR(I)),
  826    12    SJOB(I), SHOLE(I) THUS
 ****          ***        ***
  828    13  ELSE
  829    14  LET TOTCOR=TOTCOR+SJOB(I)+SHOLE(I)
  830    15  IF I NE TAIL
  831    16       LET I=SUC(I)
  832    17       GO TO NEXT
  833    18  ELSE LET CORSIZE=TOTCOR
  834    19  **
  835    20  **     COMPARE SUM OF MEMORY MAP PROGRAM AND
  836    21  **     HOLE SIZES WITH TSS CORE SIZE.
  837    22  IF TOTCOR NE T/COR PRINT 1 LINE AS FOLLOWS
 *** ERROR - MEMORY MAP DOES NOT VERIFY
  839    23  STOP
  840    24  **
  841    25  **     SCAN QUEUES FOR JOBS (1) WAITING FOR CORE,
  842    26  **     AND (2) ELIGIBLE FOR THE CPU.
  843    27  **
  844    28  ELSE LET J=0  LET TUST=UST
  845    29  FOR EVERY UST OF MMC DO
  846    30       IF FL19+FL21+FL22+FL23+FL24 EQ 0 ADD 1 TO J
  847    31       ELSE LOOP
  848    32  LET WAITCOR=J  LET J=0
  849    33  FOR EVERY UST OF CPUC DO
  850    34       IF FL18+FL19+FL21+FL23+FL34 EQ 0 AND FL22 EQ 1
  851    35            ADD 1 TO J
  852    36       ELSE LOOP
  853    37  LET ELIGCPU=J  LET UST=TUST
  854    38  RETURN END
```

```
 855    1  ''
 856    2  ''     SWAP OUT JOB
 857    3  ''
 858    4  ROUTINE SWOUT
 859    5  RDIAG
      SWOUT CALLED
 861    6  LET FL22=0
 862    7  LET FL21=1
 863    8  ADD 1 TO TSWAP
 864    9  IF FL19 EQ 1 LET FL19=1
 865   10       LET FL34=0
 866   11  ELSE CALL SWPLD(1)
 867   12  ADD 0.11 TO EXT
 868   13  RETURN END
```

```
7950T 01  08-28-75   17.504   CACI SIMSCRIPT II.5 FOR HIS 600/6000        USAF RELEASE 9.

869      1  ''
870      2  ''     SWAP IN JOB
871      3  ''
872      4  ROUTINE SWIN
873      5  PDIAG
       SWIN CALLED
875      6  LET FL23=1
876      7  CALL SWFLD(6)
877      8  ADD 1 TO INCORE
878      9  ADD 0.035 TO EXT
879     10  RETURN END
```

```
7950T 01  08-28-75   17.504   CACI SIMSCRIPT II.5 FOR HIS 600/6000      USAF RELEASE 9.

  880     1  ''
  881     2  ''     PERFORM SWAP (IN AND OUT)
  882     3  ''
  883     4  ROUTINE SWPLD(DIR)
  884     5  RDIAG
         SWPLD CALLED
  886     6  ADD 1 TO LTC32
  887     7  CALL ATCHG(1)
  888     8  ADD 1 TO SFUSE
  889     9  ADD 0.42 TO EXT
  890    10  IF DIR EQ 0 SCHEDULE AN ZALLCC GIVEN UST IN SWAPDUR MS.
  891    11       FILE THIS ZALLCC IN THE CCQ  RETURN
  892    12  ELSE SCHEDULE AN 1ALLCC GIVEN UST IN SWAPDUR MS.
  893    13  ADD LSIZE TO TSWPK
  894    14  FILE THIS 1ALLCC IN THE CCQ  RETURN END
```

```
7950T 01  08-28-75   17.504   CACI SIMSCRIPT II.5 FOR HIS 600/6000        USAF RELEASE 9.

  695     1  **
  896     2  **     TERMINATE EXECUTION OF SUBSYSTEM
  897     3  **     AND TAKE ACCOUNTING INFORMATION
  898     4  **
  899     5  ROUTINE SSFINI
  900     6  PDIAG
     SSFINI CALLED
  902     7  ADD 1 TO TKILL
  903     8  SUBTRACT 1 FROM INCORE
  904     9  CALL ATCHG(3)
  905    10  CALL MBD
  906    11  **
  907    12  **     REMOVE JOB FROM QUEUES
  908    13  IF THIS UST IS IN CPUQ REMOVE THIS UST FROM CPUQ
  909    14       IF QSWITCH NE 0 CALL COPRINT ELSE
  910    15  ELSE IF THIS UST IS IN MEMQ REMOVE THIS UST FROM MEMQ
  911    16       IF QSWITCH NE 0 CALL MOPRINT ELSE
  912    17  **
  913    18  **     KILL ANY REMAINING COURTESY CALL
  914    19  ELSE IF EK19 EQ 1
  915    20       LET I=OWNCC
  916    21       ADD 1 TO TSWAP ADD 1 TO TASIO  ADD 1 TO NOSWAP
  917    22       ADD 1 TO LTM32 ADD 1 TO TSWPK ADD SWAPDUR TO LTM1
  918    23       CANCEL THIS MIDCC CALLED I
  919    24       REMOVE THIS I FROM THE CCC
  920    25       DESTROY THIS MIDCC CALLED I
  921    26  ELSE CALL SACT  DESTROY THIS UST
  922    27  ADD 0.24 TO EXT
  923    28  RETURN END
```

```
7950T 01   08-28-75   17.504   CACI SIMSCRIPT II.5 FOR HIS 600/6000      USAF RELEASE 9.

  924     1  ''
  925     2  ''     REMOVE KEY INPUT JOBS FROM THE CPUQ
  926     3  ''
  927     4  ROUTINE BUFDMP
  928     5  RDIAG
      BUFDMP CALLED
  930     6  IF INDEPL EQ 0 ADD 0.046 TO EXT GO TO BUF.3B
  931     7  ELSE IF UST IS IN CPUQ
  932     8       REMOVE THIS UST FROM CPUQ
  933     9       LET TLFLG=1 LET MPWF=1 FILE THIS UST IN MEMQ
  934    10       CALL ATCHG(0)
  935    11  ELSE LET FLAG=1
  936    12  ADD 0.17 TO EXT
  937    13  IF OSWITCH NE 0 CALL MOPRINT CALL COPRINT ELSE
  938    14  'BUF.3' CALL KIOSRT
  939    15  RETURN END
```

```
7950T 01  08-28-75   17.504   CACI SIMSCRIPT II.5 FOR HIS 600/6000        USAF RELEASE 9.

 940     1  ''
 941     2  ''     START KEY I/O AND SCHEDULE COURTESY CALL FOR INPUT
 942     3  ''
 943     4  ROUTINE KIOSRT
 944     5  RDIAG
        KIOSRT CALLED
 946     6  ADD 0.22 TO EXT
 947     7  ''
 948     8  ''     SEE IF OUTPUT ALREADY IN PROGRESS.
 949     9  ''     IF SO, DESTROY THAT COURTESY CALL.
 950    10  IF FK19 EQ 1
 951    11     LET I=OUTCC
 952    12     CANCEL THIS KIOCC CALLED I
 953    13     REMOVE THIS 1 FROM THE CCQ
 954    14     DESTROY THIS KIOCC CALLED I
 955    15  ELSE LET FK19=1
 956    16  ''
 957    17  ''     PROCESS OUTPUT DERAIL
 958    18  IF INDDEL EQ 0
 959    19     'KIO'2' ADD 1 TO LTC22
 960    20     IF LTIN LE TIME.V
 961    21  ''
 962    22  ''     COLLECT RESPONSE TIME
 963    23        LET RESPT=TIME.V-LTIN
 964    24        LET LTYRS=LTYRS+RESPT
 965    25        LET LTIN=TIME.V
 966    26     ELSE ADD EXPONENTIAL.F(OUTMEAN,10) TO LTIN
 967    27     SCHEDULE A KIOCC GIVEN UST IN MAX.F(LTIN-TIME.V,0,0) MS.
 968    28     FILE THIS KIOCC IN THE CCQ
 969    29     LET OUTCC=KIOCC
 970    30     IF SWITCH NE 0 PRINT 1 LINE WITH JOBNO, TIME.V, LTIN THUS
     **** START OUTPUT AT ********.** UNTIL ********.**
 972    31     ELSE RETURN
 973    32  ''
 974    33  ''     PROCESS INPUT DERAIL
 975    34  ELSE ADD 1 TO LTC21
 976    35  IF LTIN LE TIME.V
 977    36  ''
 978    37  ''     COLLECT RESPONSE TIME
 979    38     LET RESPT=TIME.V-LTIN
 980    39     LET LTYRS=LTYRS+RESPT
 981    40     LET LTIN=TIME.V
 982    41  ELSE LET T=KIODUR
 983    42  SCHEDULE A KIOCC GIVEN UST IN T MS.
 984    43  IF SWITCH NE 0 PRINT 1 LINE WITH JOBNO, TIME.V, TIME.V+T THUS
     **** START INPUT AT ********.** UNTIL ********.**
 986    44  ELSE FILE THIS KIOCC IN THE CCQ
 987    45  RETURN END
```

```
7950T 01  08-28-75   17.504   CACI SIMSCRIPT II.5 FOR HIS 600/6000        USAF RELEASE 9.

 988     1  ''
 989     2  ''     START NEW UST ON ARRIVAL
 990     3  ''
 991     4  ROUTINE START
 992     5  ADD 1 TO TSTRT
 993     6  ''
 994     7  CALL CORSAMP
 995     8  ''
 996     9  ''     CREATE UST ENTITY AND ASSIGN ATTRIBUTES
 997    10  CREATE A UST
 998    11  LET LSIZE=SIZELIST LET VSIZE=LSIZE
 999    12  LET KILL=CPUSUP LET VCPUSUR=KILL LET CHK-PU=KILL
1000    13  LET DIOIAT=RINF.C LET KIIAT=RINF.C LET KOIAT=RINF.C
1001    14  LET I=NOKIC IF I GT 0
1002    15      LET DIOIAT=KILL/I LET VDIOIAT=0.5*DIOIAT
1003    16  ELSE LET I=NOKIN IF I GT 0
1004    17      LET KIIAT=KILL/I LET VKIIAT=0.5*KIIAT
1005    18  ELSE LET I=NOKOUT IF I GT 0
1006    19      LET KOIAT=KILL/I LET VKOIAT=0.5*KOIAT
1007    20  ELSE LET NXTKIN=0.5*DIOIAT LET NXTKIN=0.5*KIIAT LET NXTKOUT=0.5*KOIAT
1008    21  LET FL24=1
1009    22  LET JOBNG=TSTRT
1010    23  LET LIIN=IIME.V LET LIMAT=TIME.V
1011    24  LET TIFLG=1 LET LPWF=1 FILE THIS UST IN MEMO
1012    25  IF FVSWITCH+SSSWITCH GT 0  SHOW,JOBNO,LSIZE,KILL THUS
START   AT ********.*   OF *****  SIZE=*** CPU=********.*
1014    26  ELSE CALL ATCHG(4)
1015    27  IF OSWITCH NE 0 CALL NOPRINT ELSE
1016    28  ADD 0.13 TO EXT
1017    29  RETURN END
```

```
7950T 01  08-28-75   17.504   CACI SIMSCRIPT II.5 FOR HIS 600/6000        USAF RELEASE 9.

 1018    1  ""
 1019    2  ""     DISPATCH CPU TO SUBSYSTEM
 1020    3  ""
 1021    4  EVENT RETSSX
 1022    5  EVDIAG SHOW,JOBNO THUS
RETSSX AT ********.*   TO ****
 1024    6  ""
 1025    7  ""     SEE WHAT WILL INTERRUPT IT.
 1026    8  LET NXTCC=RINF.C
 1027    9  IF CCC IS NOT EMPTY LET NXTCC=TIME.A(F.CCC)-TIME.V+0.5 ELSE
 1028   10  LET NEXT=MIN.F(NXTDIO,NXTKIN,NXTKOUT,KILL,NXTINT,TCDEL,NXTCC)
 1029   11  LET DISPT=TIME.V
 1030   12  LET NXTDIO=NXTDIO-NEXT
 1031   13  LET NXTKIN=NXTKIN-NEXT
 1032   14  LET NXTKOUT=NXTKOUT-NEXT
 1033   15  LET KILL=KILL-NEXT
 1034   16  LET NXTINT=NXTINT-NEXT
 1035   17  IF NEXT LE .. LET NEXT=0 ELSE
 1036   18  ""
 1037   19  ""     SCHEDULE THE INTERRUPTING EVENT
 1038   20  IF NXTDIO LE 0 SCHEDULE A DRLDIO IN NEXT+0.063 MS. RETURN
 1039   21  ELSE IF NXTKIN LE 0 SCHEDULE A KONDEL IN NEXT+0.063 MS. RETURN
 1040   22  ELSE IF NXTKOUT LE 0 SCHEDULE A KOTDEL IN NEXT+0.063 MS. RETURN
 1041   23  ELSE IF KILL LE 0 SCHEDULE A DELRET IN NEXT+0.063 MS. RETURN
 1042   24  ELSE SCHEDULE A EXPNTL IN NEXT+0.063+UNIFORM.F(MNDELAY, MXDELAY,10) MS.
 1043   25  RETURN END
```

```
1044    1  **
1045    2  **     ACCUMULATE STATISTICS AFTER SUBSYSTEM TERMINATION
1046    3  **
1047    4  ROUTINE SACT
1048    5  RDIAG
       SACT CALLED
1050    6  **
1051    7  ** TALLY SUBSYSTEM STATE TIMES AND COUNTS.
1052    8  LET LTM0SS=LTM0
1053    9  LET LTM1SS=LTM1
1054   10  LET LTM2SS=LTM2
1055   11  LET LTM3SS=LTM3
1056   12  LET LTM4SS=LTM4
1057   13  LET LTM5SS=LTM5
1058   14  LET SSKIN=LTC21
1059   15  LET SSKOUT=LTC22
1060   16  LET SSDIO=LSTIO
1061   17  LET SSPSWAP=LTC31
1062   18  LET SSSWAPO=LTC32
1063   19  **
1064   20  **     CALCULATE RESPONSE TIMES.
1065   21  IF LTC21+LTC22 EQ 0
1066   22      LET LTMRS=TIME.V-LTIN LET LTC21=1 LET RESPT=LTMRS
1067   23  ELSE LET SSRESP=LTMRS/(LTC21+LTC22)
1068   24  LET SSCPU=LSITS
1069   25  IF SSSWAPCNT NE 0 PRINT 2 LINES WITH TIME.V,JOBNO,SSKIN,SSKOUT,
1070   26      LTMRS,CHPCPU,LSPTS THUS
STOP    AT ********.*   OF **** KIN=*** KOUT=***
   TOT RESP=*******.* CPU ALLOC=********.* USED=********.*
1073   27  ELSE RETURN END
```

```
7950T 01  08-28-75   17.504   CACI SIMSCRIPT II.5 FOR HIS 600/6000        USAF RELEASE 9.

1074     1  ''
1075     2  ''     UPDATE SUBSYSTEM STATE TIMES
1076     3  ''
1077     4  ROUTINE ATCHG(N)
1078     5  RDIAG
       ATCHG CALLED
1080     6  GO TO AT0,AT1,AT2,AT3,AT4 OR AT5 PER LTCH+1
1081     7  'AT0' LET LTM0=LTM0+(TIME.V-LTMWT) GO TO LT6
1082     8  'AT1' LET LTM1=LTM1+(TIME.V-LTMWT) GO TO LT6
1083     9  'AT2' LET LTM2=LTM2+(TIME.V-LTMWT) GO TO LT6
1084    10  'AT3' LET LTM3=LTM3+(TIME.V-LTMWT) GO TO LT6
1085    11  'AT4' LET LTM4=LTM4+(TIME.V-LTMWT) GO TO LT6
1086    12  'AT5' LET LTM5=LTM5+(TIME.V-LTMWT)
1087    13  'LT6' LET LTCH=N
1088    14  LET LTMWT=TIME.V
1089    15  ADD 0.11 TO EXT
1090    16  RETURN END
```

```
7950T 01  08-28-75   17.504   CACI SIMSCRIPT II.5 FOR HIS 600/6000      USAF RELEASE 9.

 1091    1  ''
 1092    2  ''     ENTRY TO ALLOCATOR BY INTERRUPT
 1093    3  ''
 1094    4  EVENT EXENTR
 1095    5  EVDIAG SHOW,JOPNO THUS
EXENTR AT ********.*   BY ****
 1096    6  REMOVE UST FROM CPUO
 1098    7  LET TLFLG=1 FILE UST LAST IN CPUO
 1099    8  IF QSWITCH NE 0 CALL COPPINT ELSE
 1100    9  CALL EXEACT
 1101   10  SCHEDULE AN ALLOCI IN 0.22 MS.
 1102   11  ''
 1103   12  ''     SAMPLE NEXT INTERRUPT TIME.
 1104   13  IF NXTINT LE 0 LET NXTINT=EXPONENTIAL.F(INTMEAN,9)
 1105   14  ELSE RETURN END
```

```
7950T 01  08-28-75   17.504   CACI SIMSCRIPT II.5 FOR HIS 600/6000        USAF RELEASE 9.

1106     1  ''
1107     2  ''     TAKE ACCOUNTING AFTER INTERRUPT
1108     3  ''
1109     4  ROUTINE EXEACT
1110     5  RDIAG
       EXEACT CALLED
1112     6  ADD 0.11 TO EXT
1113     7  ''
1114     8  ''     ACCUMULATE CPU TIME USED BY SUBSYSTEM AND TSS.
1115     9  ADD 1 TO TAG'C
1116    10  ADD NEXT TO LSPTS
1117    11  ADD NEXT TO TAGPT
1118    12  ADD NEXT TO LASTU
1119    13  IF TIME.V-PAST" GE ASD3I LET MPWF=1
1120    14  ELSE IF TIME.V-TLLST GE TLTLY LET LSFLG=1
1121    15  ELSE RETURN END
```

```
7950T 01  08-28-75   17.504   CACI SIMSCRIPT II.5 FOR HIS 600/6000        USAF RELEASE 9.

 1122     1  ''
 1123     2  ''     KEY INPUT DETAIL
 1124     3  ''
 1125     4  EVENT KONDRL
 1126     5  FVDIAG SHOW,JOBNO THUS
KONDRL AT ********.*   BY ****
 1128     6  LET INDDRL=1
 1129     7  CALL EXEACT
 113C     8  ADD 1 TO KEYIN
 1131     9  CALL BUFDMP
 1132    10  ''
 1133    11  ''     SAMPLE NEXT INPUT TIME
 1134    12  LET NXTKIN=KIIAT LET VKIIAT=NXTKIN
 1135    13  SCHEDULE AN ALLOCI IN 0.2 MS.
 1136    14  RETURN END
```

```
7950T 01  08-28-75   17.504   CACI SIMSCRIPT II.5 FOR HIS 600/6000      USAF RELEASE 9,

 1137     1  **
 1138     2  **     KEY OUTPUT DETAIL
 1139     3  **
 1140     4  EVENT KOTDRL
 1141     5  EVDIAG SHOW,JORNO THUS
KOTDRL AT ********.*  BY ****
 1143     6  LET INDDRL=0
 1144     7  CALL EXFACT
 1145     8  ADD 1 TO KEYOUT
 1146     9  CALL BUFDMP
 1147    10  **
 1148    11  **     SAMPLE NEXT OUTPUT TIME
 1149    12  LET NXTKOUT=KOIAT LET VKOIAT=NXTKOUT
 1150    13  SCHEDULE AN ALLOCI IN C.22 MS,
 1151    14  RETURN END
```

```
7950T 01  08-28-75   17.504   CACI SIMSCRIPT II.5 FOR HIS 600/6000      USAF RELEASE 9.

 1152     1  **
 1153     2  **    CERAIL TO PERFORM DISK I/O
 1154     3  **
 1155     4  EVENT DRLDIO
 1156     5  EVDIAG SHOW,JOENO THUS
DRLDIO AT ********.*   BY ****
 1158     6  LET FL18=1
 1159     7  CALL EXEACT
 1160     8  ADD 1 TO DISKIO
 1161     9  ADD 1 TO LSTIO
 1162    10  LET NXTDIO=DIOIAT LET VDIOIAT=NXTDIO
 1163    11  **
 1164    12  **    FILE REQUEST IN DIOC. IF DIOQ WAS EMPTY
 1165    13  **    START -IO NOW AND SCHEDULE DIC COURTESY CALL.
 1166    14  IF DIOQ IS EMPTY
 1167    15     SCHEDULE A DIOCC GIVEN UST IN EXPONENTIAL.F(DIOMEAN,10) MS.
 1168    16     FILE THIS DIOCC IN THE CCQ
 1169    17  ELSE FILE THIS UST LAST IN DIOQ
 1170    18  SCHEDULE AN ALLOCI IN 0.35 MS.
 1171    19  RETURN END
```

```
79507 01  08-28-75   17.504   CACI SIMSCRIPT II.5 FOR HIS 600/6000        USAF RELEASE 9.

 1172     1  **
 1173     2  **     DERAIL TO TERMINATE SUBSYSTEM
 1174     3  **
 1175     4  EVENT DRLRET
 1176     5  EVDIAG SHOW,JOBNO THUS
DRLRET AT ********.*   BY ****
 1178     6  CALL EXEACT
 1179     7  **
 1180     8  **     CHECK CPU USAGE OF SUBSYSTEM.
 1181     9  IF ABS.F(CHKCPU-LSPTS) GT 1
 1182    10       PRINT 1 LINE WITH JOBNO, CHKCPU, LSPTS THUS
*** ERROR - **** ALLOCATED CPU=******.* USED ******.*
 1184    11  ELSE CALL SSPINI
 1185    12  SCHEDULE AN ALLOCI IN 0.176 MS.
 1186    13  RETURN END
```

```
7950T 01  08-28-75   17.504   CACI SIMSCRIPT II.5 FOR HIS 600/6000        USAF RELEASE 9.

 1187     1  ''
 1188     2  ''     KEY INPUT AND OUTPUT COURTESY CALLS
 1189     3  ''
 1190     4  EVENT KIOCC SAVING THE EVENT NOTICE
 1191     5  LET TUST=USE LET UST=KUST
 1192     6  EVDIAG SHOW,JOBNO THUS
KIOCC  AT ********.*  FOR ****
 1193     7  IF KSWITCH NE 0 PRINT 1 LINE WITH JOBNO, TIME.V THUS
    **** FINISHED KIO AT ********.**
 1196     8  ELSE LET LTIN=TIME.V
 1197     9  LET FK19=0
 1198    10  ''
 1199    11  ''     SEE IF IT WAS AN OUTPUT
 1200    12  IF FL19 EQ 0
 1201    13      GO TO KIOCC.1
 1202    14  ELSE LET FL19=0
 1203    15  IF FL21 EQ 0
 1204    16      IF FL22 EQ 1
 1205    17          REMOVE THIS UST FROM MSMQ
 1206    18          LET FLFLG=1 FILE THIS UST FIRST IN CPUQ
 1207    19          CALL ATCHG(2)
 1208    20          IF CSWITCH NE 0 CALL MQPRINT CALL COPRINT ELSE
 1209    21          GO TO KIOCC.1
 1210    22      ELSE LET MPWF=1
 1211    23      CALL ATCHG(4)
 1212    24  ELSE
 1213    25  'KIOCC.1' LET UST=TUST REMOVE THIS KIOCC FROM THE CCQ
 1214    26  DESTROY THIS KIOCC
 1215    27  RETURN END
```

```
7950T 01   08-28-75   17.504   CACI SIMSCRIPT II.5 FOR HIS 600/6000        USAF RELEASE 9.

 1216     1  ''
 1217     2  ''     DISK I/O COURTESY CALL
 1218     3  ''
 1219     4  EVENT DIOCC SAVING THE EVENT NOTICE
 1220     5  LET TUST=UST LET UST=DUST
 1221     6  EVDIAG SHOW,JOENO THUS
DIOCC   AT ********.*   OF ****
 1223     7  REMOVE FIRST UST FROM DIOQ
 1224     8  LET FL18=)
 1225     9  ''
 1226    10  ''     MOVE HIM TO TOP OF THE CPUQ.
 1227    11  REMOVE THIS UST FROM THE CPUQ
 1228    12  LET TLFLG=1 FILE THIS UST FIRST IN THE CPUQ
 1229    13  IF OSWITCH AT 1 CALL COPRINT ELSE
 1230    14  LET UST=TUST
 1231    15  ''
 1232    16  ''     SEE IF ANY MORE DISK I/O'S ARE WAITING.
 1233    17  REMOVE THIS DIOCC FROM THE CCQ
 1234    18  IF DIOQ IS NOT EMPTY
 1235    19     RESCHEDULE THIS DIOCC GIVEN F.DIOQ IN EXPONENTIAL.F(DIOMEAN,10) MS.
 1236    20     FILE THIS DIOCC IN THE CCQ
 1237    21     RETURN
 1238    22  ELSE DESTROY THIS DIOCC
 1239    23  RETURN END
```

```
7950T 01  08-28-75   17.504   CACI SIMSCRIPT II.5 FOR HIS 600/6000        USAF RELEASE 9.

1240    1  ''
1241    2  ''     SWAP OUT COURTESY CALL
1242    3  ''
1243    4  EVENT 1ALLCC SAVING THE EVENT NOTICE
1244    5  LET TUST=UST LET UST=SUST1
1245    6  EVDIAG SHOW,JOBNO THUS
1ALLCC AT ********.*   OF ****
1247    7  SUBTRACT 1 FROM INCORE
1248    8  LET FL21=)
1249    9  LET MPWF=1
1250   10  ''     SEE WHAT HE GOT SWAPPED OUT FOR.
1251   11  IF FL19 EQ 2
1252   12      IF FL34 EQ 1
1253   13          CALL ATCHG(5)
1254   14          GO TO ALLCCA
1255   15      ELSE CALL ATCHG(4)
1256   16       GO TO ALLCCA
1257   17  ELSE ADD 1 TO IASIO
1258   18  CALL ATCHG(3)
1259   19  'ALLCCA' CALL MOD
1260   20  SUBTRACT 1 FROM SFUSE
1261   21  LET UST=TUST REMOVE THIS 1ALLCC FROM THE CCQ
1262   22  DESTROY THIS 1ALLCC
1263   23  RETURN END
```

```
7950T 01  08-28-75   17.504  CACI SIMSCRIPT II.5 FOR HIS 600/6000      USAF RELEASE 9.

1264     1  **
1265     2  **     SWAP IN COURTEST CALL
1266     3  **
1267     4  EVENT 2ALLCC SAVING THE EVENT NOTICE
1268     5  LET TUST=US. LET UST=SUST2
1269     6  EVDIAG SHOW,JOENO THUS
2ALLCC AT ********.*    OF ****
1271     7  LET FL22=1
1272     8  LET FL23=0
1273     9  LET FL34=0
1274    10  REMOVE THIS UST FROM MEMQ
1275    11  LET TLFLG=1 FILE THIS UST FIRST IN THE CPUQ
1276    12  CALL ARCHG(2)
1277    13  IF QSWITCH NE 1 CALL XOFRINT CALL CLPRINT ELSE
1278    14  SUBTRACT 1 FROM SPUSE
1279    15  LET UST=TUST REMOVE THIS 2ALLCC FROM THE CCQ
1280    16  DESTROY THIS 2ALLCC
1281    17  RETURN END
```

```
70501 01  08-28-75   17.504   CACI SIMSCRIPT II.5 FOR HIS 600/6000        USAF RELEASE 9.

 1282     1  **
 1283     2  **     LINE SERVICE
 1284     3  **
 1285     4  EVENT LINSV
 1286     5  EVDIAG SHOW THUS
LINSV AT ********.*
 1285     6  LET LSFLG=0
 1286     7  LET MSR240=0
 1290     8  LET TLLST=TIME.V
 1291     9  LET E/T=0.03
 1292    10  IF TIME.V-TLOLD LT TLNLM GO TO MSRK
 1293    11  ELSE LET TLOLD=TIME.V
 1294    12  **     SEE IF TSS IS IDLE
 1295    13  IF MEMQ HS SHPOL AND CPUQ IS EMPTY LET MSR240=1
 1296    14      IF TACOR NE TASMS GO TO MSRK5
 1297    15      ELSE GO TO MSR300
 1298    16  ELSE
 1299    17  **
 1300    18  **
 1301    19  'MSRK' LET MSR240=0
 1302    20  'MSRK5' IF TIME.V-TASRT LT TAMRI GO TO MSR300
 1303    21  **
 1304    22  **     CALCULATE % CORE UTILIZATION TO SEE IF TSS
 1305    23  **     CORE SIZE SHOULD BE REDUCED
 1306    24  ELSE LET I=HTAL LET TEMP=0
 1307    25  'LOOP' LET TEMP=TEMP+SJOB(I) LET I=SUC(I)
 1308    26  IF I NE 0 GO TO LOOP
 1309    27  ELSE LET TAP U=(((TEMP*100)/TACOR)+TAPMU)/2
 1310    28  **
 1311    29  **     FIND MAPCOST PROGRAM I. SYSTEM.
 1312    30  LET TALPS=0
 1313    31  FOR EVERY UST OF MEMQ DO
 1314    32  IF LSIZE GT TALPS LET TALPS=LSIZE
 1315    33  ELSE LOOP
 1316    34  FOR EVERY UST OF CPUQ DO
 1317    35  IF LSIZE GT TALPS LET TALPS=LSIZE
 1318    36  ELSE LOOP
 1319    37  IF MSR240 EQ 1 GO TO MSR250
 1320    38  ELSE IF TIME.V-TLSRT LT TAMRI GO TO MSR300
 1321    39  ELSE LET TASRT=TIME.V
 1322    40  IF ALUTH GE TI B.V-TAMRI GO TO MSR300
 1323    41  ELSE IF TIME.V-TALCT LT TATIM GO TO MSR300
 1324    42  ELSE IF TATIM EQ 0 GO TO MSR300
 1325    43  ELSE IF TATIM EQ TAPMP GO TO MSR300
 1326    44  ELSE IF MSR25 = IF TACOR-TASRI GT TASMS LET CHANGE=TACOR-TASRI
 1327    45  ELSE IF TACOR-TASRT LE TASMS LET CHANGE=TASMS
 1328    46  ELSE IF TAMOL NE 0 GO TO MSR300
 1329    47  ELSE IF CHANGE LT TALPS GO TO MSR300
 1330    48  ELSE IF TIME.V-TALCT LT TASCP GO TO MSR300
 1331    49  ELSE LET MPACT=1
 1332    50  LET TAMOL=CHANGE
 1333    51  ADD 1 TO TSRRC
```

```
79SCT 01  08-29-75   17.504   CACI SIMSCRIPT II.5 FOR HIS 600/6000      USAF RELEASE 9.

1334    52  ADD 0.44 TO EXT
1335    53  ''
1336    54  ''      SEE IF ANY NEW SUBSYSTEMS (UST'S) NEED TO
1337    55  ''              BE STARTED.
1338    56  ''
1339    57  'MSR3C0' IF NXTUST EQ TIME.V CALL START
1340    58      LET VUSTIAT=USTIAT
1341    59      LET NXTUST=NXTUST+VUSTIAT
1342    60      GO TO MSR3C0
1343    61  ELSE IF TLFLG+MPWF GE 1 SCHEDULE AN ALLOCI IN EXT MS. RETURN
1344    62  ''
1345    63  ''      TSS IS IDLE,,,,,,,,
1346    64  ''      RELINQUISH TO GPSS UNTIL NEXT INTERRUPT,
1347    65  ''      SUBSYSTEM APPIAL, OR COURTESY CALL.
1348    66  ELSE ADD 1 TO TLMAA LET TAGPT=0
1349    67  IF TIME.V=TAPT OR A3D3A LET MPWF=1
1350    68  ELSE LET NXTCC=RINF.C
1351    69  IF CCG IS NOT EMPTY LET NXTCC=TIME.A(F,CCG)
1352    70  ELSE LET WAKET=MIN.F(NXTUST+0.001,TIME.V+TAGMI,NXTCC+0.16)
1353    71  SCHEDULE AN ALLOCI AT WAKET+EXT
1354    72  RETURN END
```

References

1. Bauer,M.F., and Irani, K.B.; "A Simulation Model of TSS 8.1," RADC Technical Report, 1975, F30602-73-C-0001.

2. "Time-Sharing System Executive SMD ," (BR29, Rev.2), Honeywell Information Systems Inc., 1974.

3. Hahn, T.; "Type 19 Accounting Records : Data Reduction and Analysis Programs," Systems Engineering Laboratory, University of Michigan, 1975.

4. Mulla, J.D.; "TSS Data Reduction and Analysis Programs for TSS Release 8.1," Systems Engineering Laboratory, University of Michigan, 1975.

5. "Simscript II.5 User's Manual, Honeywell 600/6000 Version," Consolidated Analysis Centers Inc., 1972.

6. Kiviat, P.J., Villanueva, R., Markowitz, H.M.; "The Simscript II Programming Language," Prentice-Hall, Inc., 1968.

MISSION
of
Rome Air Development Center

RADC is the principal AFSC organization charged with planning and executing the USAF exploratory and advanced development programs for information sciences, intelligence, command, control and communications technology, products and services oriented to the needs of the USAF. Primary RADC mission areas are communications, electromagnetic guidance and control, surveillance of ground and aerospace objects, intelligence data collection and handling, information system technology, and electronic reliability, maintainability and compatibility. RADC has mission responsibility as assigned by AFSC for demonstration and acquisition of selected subsystems and systems in the intelligence, mapping, charting, command, control and communications areas.